**SKRIPSI – ME 141501**

# ANALISA PERUBAHAN WARNA HSV PADA PENGOLAHAN CITRA TERHADAP INTENSITAS CAHAYA SEBAGAI DASAR PENERAPAN MASUKAN KONTROL *AUTOMATIC STACKING CRANE*

MUHAMMAD BAHRU SHOLAHUDDIN
NRP 4213 100 077

Dosen Pembimbing
Dr. Ir. A. A. Masroeri, M. Eng
Juniarko Prananda, S.T., M.T.

DEPARTEMEN TEKNIK SISTEM PERKAPALAN
Fakultas Teknologi Kelautan
Institut Teknologi Sepuluh Nopember
Surabaya
2017

*Halaman ini sengaja dikosongkan*.

**FINAL PROJECT – ME 141501**

# ANALYSIS OF HSV COLOR MODEL CHANGE IN IMAGE PROCESSING TOWARD LIGHT INTENSITY AS BASIC OF CONTROL INPUT IN AUTOMATIC STACKING CRANE

MUHAMMAD BAHRU SHOLAHUDDIN
NRP 4213 100 077

Advisor Lecturer
Dr. Ir. A. A. Masroeri, M. Eng
Juniarko Prananda, S.T., M.T.

DEPARTMENT OF MARINE ENGINEERING
Faculty of Marine Technology
Institut Teknologi Sepuluh Nopember
Surabaya
2017

*Halaman ini sengaja dikosongkan.*

**LEMBAR PENGESAHAN**

# ANALISA PERUBAHAN WARNA HSV PADA PENGOLAHAN CITRA TERHADAP INTENSITAS CAHAYA SEBAGAI DASAR PENERAPAN MASUKAN KONTROL *AUTOMATIC STACKING CRANE*

**TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Teknik
pada
Bidang Studi *Marine Electrical and Automation System* (MEAS)
Program Studi S-1 Departemen Teknik Sistem Perkapalan
Fakultas Teknologi Kelautan
Institut Teknologi Sepuluh Nopember

Oleh:
**Muhammad Bahru Sholahuddin**
NRP 4213 100 077

Disetujui oleh Pembimbing Tugas Akhir:

1. Dr. Ir. A. A Masroeri, M. Eng ( )
2. Juniarko Prananda, S.T., M.T. ( )

SURABAYA
Juli, 2017

*Halaman ini sengaja dikosongkan.*

## LEMBAR PENGESAHAN

# ANALISA PERUBAHAN WARNA HSV PADA PENGOLAHAN CITRA TERHADAP INTENSITAS CAHAYA SEBAGAI DASAR PENERAPAN MASUKAN KONTROL *AUTOMATIC STACKING CRANE*

**TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Teknik
pada
Bidang Studi *Marine Electrical and Automation System* (MEAS)
Program Studi S-1 Departemen Teknik Sistem Perkapalan
Fakultas Teknologi Kelautan
Institut Teknologi Sepuluh Nopember

Oleh:
**Muhammad Bahru Sholahuddin**
NRP 4213 100 077

Disetujui oleh Kepala Departemen Teknik Sistem Perkapalan:

Dr. Eng. M. Badrus Zaman, S.T., M.T.
NIP. 19770802200801100 7

*Halaman ini sengaja dikosongkan.*

**ANALISA PERUBAHAN WARNA HSV PADA PENGOLAHAN CITRA TERHADAP INTENSITAS CAHAYA SEBAGAI DASAR PENERAPAN MASUKAN KONTROL *AUTOMATIC STACKING CRANE***


| | |
|---|---|
| **Nama** | **: Muhammad Bahru Sholahuddin** |
| **NRP** | **: 4213 100 077** |
| **Departemen** | **: Teknik Sistem Perkapalan** |
| **Dosen Pembimbing I** | **: Dr. Ir. A. A. Masroeri, M.Eng** |
| **Dosen Pembimbing II** | **: Juniarko Prananda, S.T., M.T.** |



## Abstrak

Proses dalam penanganan peti kemas diusahakan untuk bekerja secara efisien, handal, dan aman. Hal tersebut dilakukan untuk menekan biaya, dan menghindari resiko. Untuk mendukung tujuan tersebut, baik peralatan maupun proses manajemen yang berkaitan, telah mengalami banyak perkembangan. Salah satu peralatan penting dalam proses ini ialah crane. Saat ini teknologi pada crane peti kemas telah berkembang pesat, misalnya pada Automatic Stacking Crane yang mana hampir seluruh pekerjaan dapat dilakukan secara otomatis. Terdapat berbagai macam cara agar sistem kontrol mampu bekerja otomatis. Dari segi perangkat masukan, untuk mengatur posisi peti kemas saat mendarat ASC 'konecrane' menggunakan laser dan untuk mendeteksi *swaying* ataupun *skewing* menggunakan sensor infra-merah. Pada tugas akhir ini ditawarkan cara lain untuk mendeteksi object maupun gerakan pada crane, yaitu menggunakan *computer vision*. Cara ini diperkirakan mampu beroperasi lebih ekonomis, karena dapat bekerja hanya dengan satu perangkat masukan namun mampu bekerja untuk banyak pekerjaan. Namun, agar dapat diterapkan di lapangan, sebagai langkah awal perlu dianalisa terkait kemampuan sistem ini terhadap perubahan cahaya. Karena faktor cahaya merupakan salah satu tantangan dalam dunia *computer vision*, nilai warna obyek akan berubah jika intensitas cahaya yang mengenai obyek tersebut juga berubah. Dari hasil analisa, diketahui bahwa dari perubahan intensitas cahaya terhadap model warna HSV (Hue, Saturation, Value) dalam pengolahan citra, semakin besar nilai intensitas cahaya yang diperoleh sensor kamera maka semakin kecil nilai Saturation, dan semakin besar nilai Value. Hal ini ditunjukkan oleh fungsi $y = 0.0004885*e^{(0.06007*x)}$ yang mana $x$ merupakan nilai Value (HSV) rata-rata dari seluruh pixel, dan $y$ merupakan intensitas cahaya. Dan juga fungsi $y = 0.0000055556*e^{(0.07802*(255-x))}$, yang mana $x$ merupakan nilai Saturation (HSV) rata-rata dari seluruh pixel, dan $y$ merupakan intensitas cahaya. Selain itu pada tugas akhir ini dilakukan simulasi pendeteksian gerakan pada spreader menggunakan pengolahan citra.



**Kata Kunci:** *Automatic Stacking Crane*, *Image Processing*, *Computer Vision*, *Anti-sway*, OpenCV, model warna HSV

*Halaman ini sengaja dikosongkan.*

***ANALYSIS OF HSV COLOR MODEL CHANGE IN IMAGE PROCESSING TOWARD LIGHT INTENSITY AS BASIC OF CONTROL INPUT IN AUTOMATIC STACKING CRANE***


| | | |
|---|---|---|
| ***Name*** | **:** | **Muhammad Bahru Sholahuddin** |
| **NRP** | **:** | **4213 100 077** |
| ***Department*** | **:** | **Teknik Sistem Perkapalan** |
| ***Advisor Lecturer I*** | **:** | **Dr. Ir. A. A. Masroeri, M.Eng** |
| ***Advisor Lecturer II*** | **:** | **Juniarko Prananda, S.T., M.T.** |



## *Abstract*

*Container handling process need to be efficient, reliable, and safe. Those should be achieved in order to reduce operating costs and to avoid risks. Supporting those goals, either related equipment or management process has been developed a lot. One of the important tools is crane which used to transferring and stacking the containers. Nowadays crane technology has been developed rapidly, such as Automatic Stacking Crane which most of the job can be done automatically. There are several ways to implement the control system. From the input parameter side, ASC can be equipped with laser for positioning task, and infra-red for sway or skew detector, just as 'konecranes' did in Teluk Lamong. This bachelor thesis proposes another way to get the job, it is by computer vision. This way predicted to be more economics because only with one camera, it can detect many objects and movements. But before this way being brought into reality, there are challenges to solve with, one of them is related to the ambient lightness. The color value that camera see being changed as the light change. From analysis results in this bachelor thesis, when the light intensity increases, Saturation of HSV turns out to be decreases, as Value of HSV being increases. The correlation being approximated with function y = 0.0004885*e^(0.06007*x), which y is light intensity and x is average Value of HSV from all pixels. And function y = 0.0000055556*e^(0.07802*(255-x)), which y is light intensity and x is average Saturation of HSV from all pixels. This bachelor thesis also provide simulation about how the spreader motion can be detected.*



***Keywords:*** Automatic Stacking Crane, Image Processing, Computer Vision, Anti-sway, OpenCV, HSV color model

*Halaman ini sengaja dikosongkan.*

# KATA PENGANTAR

Segala puji penulis panjatkan atas kehadirat Allah SWT yang telah memberikan rahmat, taufik, inayah, serta hidayah-Nya sehingga penulis dapat mengerjakan dan menyelesaikan laporan tugas akhir dengan judul "Analisa Perubahan Warna Hsv Pada Pengolahan Citra Terhadap Intensitas Cahaya Sebagai Dasar Penerapan Masukan Kontrol *Automatic Stacking Crane*". Sholawat serta salam penulis haturkan kepada nabi Muhammad SAW yang telah menyampaikan risalah-Nya dan menjadi suri tauladan bagi umat.

Dalam penulisan tugas akhir ini, banyak sekali pihak yang telah membantu sehingga penyusunannya bisa selesai. Karena itu, penulis mengucapkan terima kasih kepada:

1. Ibu Luluk Alifah, S. E., Bapak Sohib Romdhoni, S. E., selaku orang tua yang telah banyak berkorban dan senantiasa mendukung anaknya untuk menjadi insan yang lebih baik. Serta saudari Humaida Hafidzah Zahra selaku adik kandung.
2. Bapak Dr. Ir. A. A. Masroeri, M. Eng. selaku dosen pembimbing pertama yang telah bersedia meluangkan waktunya untuk menelaah progress penulis dan memberikan fasilitas untuk mendukung kegiatan belajar.
3. Bapak Juniarko Prananda, S.T., M.T. selaku dosen pembimbing kedua yang telah bersedia meluangkan waktunya untuk berdiskusi.
4. Bapak Ir. Sardono Sarwito, M. Sc. selaku kepala laboratorium Marine Electrical and Automation System yang telah memberikan fasilitas untuk mendukung kegiatan pembelajaran.
5. Bapak Dr. Eng. M. Badrus Zaman, S.T., M.T. selaku Ketua Jurusan Teknik Sistem Perkapalan, Institut Teknologi Sepuluh Nopember dan dosen wali.
6. Mas Arma N selaku teknisi pada *Automatic Crane* Konecranes di Terminal Teluk Lamong, yang telah bersedia diwawancarai.
7. Bapak Rudy Dikairono, S.T., M.T. dan Ir. H. Agoes Santoso, M.Sc., M. Phil selaku pembimbing Barunastra Roboboat dan teman-teman tim Darwin (2013), Eric (2013), Yohan (2013), Anas (2013), Ujang (2013), Eky (2013), Topo (2013) Dwiky (2012), Rahmat (2012), Ghofur (2012), Welderson (2011), dll yang telah membagikan kesan dan pengalaman bersama.
8. Keluarga besar bani H. Achwan Affandi dan H. Abduh Rahman yang telah banyak memberikan dukungan do'a dan motivasi.
9. Para guru, asatidz, dan dosen dari TK sampai S1 yang telah memberikan ilmu dan pengalaman berguna serta do'a dan dukungan.
10. Keluarga K. H. Abdus Syakur dan Drs. Mahmud Musta'in, M. Sc, Ph. D yang telah berkenan memberikan bekal agama selama penulis bertempat di Surabaya.
11. Teman-teman pembimbingan bapak Dr. Ir. A. A. Masroeri, M. Eng dan penghuni NASDEC, Danuja (2013), Mitha (2013), Fathia (2013), Irma (2013), Naufal (2013), Adi (2013), Ali (2013), Libry (2012), dll yang telah memfasilitasi proses percetakan dokumen, diskusi, perbaikan struktur penulisan, pengayaan pustaka, serta berbagi pengalaman dan kesenangan.
12. Teman-teman MEAS, Barakuda, Kontrakan Abah, Uenak, Darussalam Keputih, Blok T-45, Pramuka Purwakirana, Finger Ar-Rahmat, Fighter SMT, Siskal ITS,

Marine Icon, UKM Maritime Challenge, UKM Robotika yang namanya tidak cukup untuk dituliskan satu per satu, dan yang telah mengingatkan, memberikan kesenangan, dukungan dan pengalaman bersama.

13. Mas Muhammad Arif Pradana selaku mentor pada kegiatan kerohanian ITS, beserta teman-teman Bowo (2013), Darori (2013), Ivan (2013), Oky (2013), Yusuf (2013), Faisal (2013), Arief Rahman (2013), Arief Maulana (2013), Kafin (2013) Miranto (2013), Manas (2013), Naufal (2013), Rhama (2013), Rizky (2013), Riko (2013), Bikso (2013), Hamzah F (2013), Heri (2013), Andhy W. (2013), Fatekhun (2013), Ifa (2013), M. A. Ghufron A. M. (2013) yang telah bersedia untuk berbagi kesenangan dan saling mengingatkan baik dalam urusan agama maupun penugasan akademik.
14. Semua pihak yang telah membantu, baik secara langsung maupun tidak langsung.

Semoga Allah SWT senantiasa memberi rahmat bagi semua pihak yang telah membantu penulis.

Karena keterbatasan yang dimiliki, penulis menyadari masih banyak sekali kekurangan pada penulisan tugas akhir ini. Oleh karena itu, penulis mengharapkan kritik dan saran agar tugas akhir ini menjadi lebih baik dan berguna dimasa yang akan datang. Harapan penulis, tugas akhir yang sederhana ini dapat bermanfaat bagi pembaca, rekan-rekan mahasiswa dan dosen-dosen di Institut Teknologi Sepuluh Nopember.

Dengan segala kerendahan hati, penulis memohon maaf atas kesalahan dalam ucapan maupun perbuatan dan atas segala perhatian diucapkan terima kasih.

Surabaya, Juli 2017

Penulis
(Muhammad Bahru Sholahuddin)

# DAFTAR ISI

# DAFTAR GAMBAR

# BAB I
# PENDAHULUAN

## 1.1. Latar Belakang

Sekarang ini kegiatan yang dilakukan oleh manusia telah banyak digantikan oleh mesin. Meski tidak sempurna seperti apa yang dimiliki oleh manusia (mata, otak, telinga, hidung, kulit, tangan, kaki, dll), mesin mampu bekerja lebih efisien karena berbagai faktor. Kamera yang berguna sebagai masukan yang dapat dianalogikan sebagai mata dan sirkuit pemroses yang dianalogikan seperti otak masih memiliki banyak kekurangan. Jika dibandingkan dengan mata, densitas titik-titik warna (pixel) pada kamera masih memiliki perbandingan nilai yang jauh. Begitu pula dengan sirkuit pemroses yang butuh masukan parameter tertentu agar proses dapat berjalan sesuai keinginan pengguna.

Sebagai pemrogram Computer Vision pada tim Roboboat Barunastra ITS, didapat permasalahan yang perlu diatasi dalam pengolahan citra. Pada kondisi terbuka dimana ada pengaruh cahaya matahari, robot kapal yang menerapkan pengolahan citra dengan segmentasi menggunakan *thresholding* warna HSV tidak mampu mendeteksi halangan dengan baik jika kondisi cahaya berubah dari sebelumnya. Hal ini disebabkan nilai warna dari halangan yang telah didapat telah berubah akibat perubahan paparan intensitas cahaya pada objek halangan tersebut. Masalah ini sebelumya diatasi dengan memperlebar rentang nilai Value pada HSV yang merupakan representasi dari intensitas pencahayaan dan melakukan kalibrasi ulang jika intensitas cahaya berubah drastis. Namun cara ini rentan karena semakin lebar rentang suatu nilai maka kemungkinan *noise* yang di dapat juga semakin besar, dan pada praktiknya nilai Saturation juga berubah. Hal ini tentunya dapat mengurangi efisiensi waktu tim dalam berkegiatan. Pada skripsi ini ingin diatasi permasalahan tersebut dengan menerapkan fungsi yang menghubungkan perubahan intensitas cahaya dengan perubahan nilai warna pada sistem pengolahan citra, sehingga sistem dapat melakukan koreksi terhadap intensitas cahaya tanpa adanya pengaturan oleh manusia. Device yang digunakan dalam proses pengerjaan skripsi ini ialah ponsel android dan komputer. Hal ini ditentukan dari pertimbangan portabilitas, kompabilitas, dan ekonomis. Pada pengambilan data digunakan device android karena mudah untuk dipergunakan dan sudah terintegrasi antara sirkuit pemroses, display interface, kamera, dan sensor lain.

Dalam kaitannya dengan bidang teknologi kelautan, hasil analisa ini dapat diterapkan baik dalam monitoring ataupun kontrol suatu alat. Sebagai contoh, dalam hal monitoring Computer Vision telah diterapkan dalam membantu kegiatan Remote Sensing. Dalam hal kontrol alat, Computer Vision dapat diterapkan dalam Robot yang berkaitan, seperti halnya robot kapal, Automatic Crane, dll. Penerapan yang ditawarkan dalam skripsi ini ialah pada Automatic Stacking Crane dimana sistem pengolahan citra dapat digunakan untuk mengidentifikasi spreader, container, dan gantry serta memperkirakan sifat dari objek tersebut seperti posisi, arah gerakan, dan kecepatan. Namun pada skripsi ini, hasil analisa yang dilakukan berdasar warna hanya dipergunakan untuk mengidentifikasi spreader. Sedangkan untuk pengidentifikasian dengan cara lain

(misal analisa bentuk berdasarkan pola antar pixel) atau untuk objek lain dilakukan pada penelitian lain.

## 1.2. Perumusan Masalah

1. Bagaimana respon perubahan nilai HSV objek terhadap perubahan intensitas cahaya?
2. Bagaimana penerapan kontrol pengolahan citra terhadap warna dengan kondisi cahaya yang berubah?

## 1.3. Batasan Masalah

1. Proses pengolahan citra gambar pada kegiatan ini dilakukan hanya pada bidang matrix 2 dimensi.
2. Proses pengolahan berdasarkan respon umum, tidak diperhitungkan tingkat reflektifitas cahaya material.
3. Objek percobaan berupa benda datar.

## 1.4. Tujuan

1. Mendapatkan respon umum dari perubahan nilai HSV objek terhadap perubahan intensitas cahaya.
2. Mensimulasikan penerapan pengolahan citra dari hasil analisa dan percobaan yang telah dilakukan

## 1.5. Manfaat

1. Menyediakan data karakteristik nilai warna HSV suatu objek terhadap intensitas cahaya.
2. Meningkatkan kinerja sistem yang menerapkan Computer Vision 2D pada kondisi yang berubah-ubah kondisi cahayanya (kondisi terbuka).

# BAB II
# TINJAUAN PUSTAKA

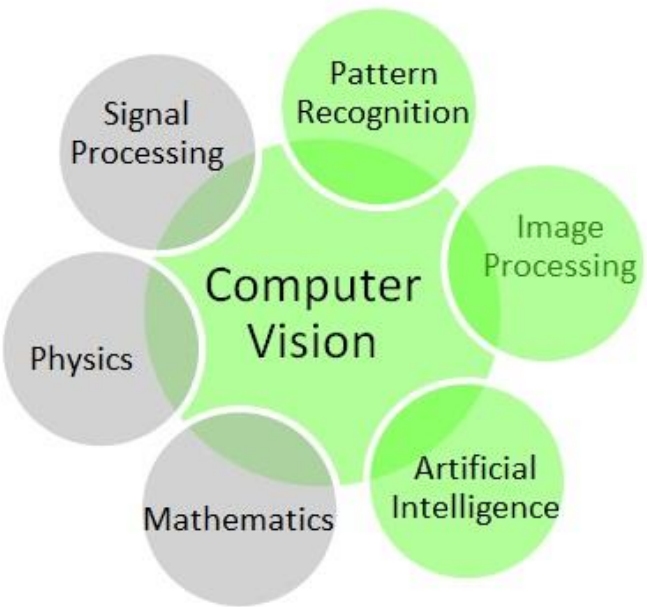
## 1.1. Pengolahan Citra (*Image Processing*)

Pengolahan citra adalah pengolahan gambar, rangkaian gambar atau video, seperti foto atau video. Output dari pengolahan citra bisa berupa gambar atau sekumpulan karakteristik atau parameter yang berhubungan dengan gambar. [1] Sebagian besar teknik pengolahan citra bekerja dengan cara mengisolasi sinyal dua dimensi dan menerapkan pemrosesan sinyal pada ggambar. Gambar juga diproses sebagai sinyal tiga dimensi dengan dimensi ketiga berupa waktu ataupun sumbu-z.

Pengolahan citra umumnya mengacu pada pengolahan citra digital, namun pemrosesan gambar optik dan analog juga bisa masuk kategori ini. Akuisisi gambar (mendapatkan gambar masukan pada wadah pertama) juga dapat disebut sebagai pencitraan. [2]

Bidang yang berkaitan erat dengan pengolahan citra adalah komputer grafis dan *computer vision*. Dalam komputer grafis, gambar dibuat secara manual dari model fisik objek, lingkungan, dan pencahayaan, tidak diperoleh melalui perangkat pencitraan seperti kamera dari pemandangan alami. Sedangkan *Computer vision*, sering dianggap sebagai pengolahan citra tingkat tinggi dimana mesin / komputer / perangkat lunak dapat menguraikan dan menjelaskan isi fisik gambar atau urutan gambar. Hubungan pengolahan citra dengan *computer vision* ditunjukkan dalam Gambar 2.1.

Dalam ilmu pengetahuan dan teknologi modern, gambar juga mendapatkan cakupan yang jauh lebih luas karena semakin pentingnya visualisasi ilmiah (seringkali data ilmiah atau eksperimen kompleks berskala besar). Contohnya termasuk data microarray dalam penelitian genetika, atau perdagangan portofolio multi-aset real-time di bidang keuangan.



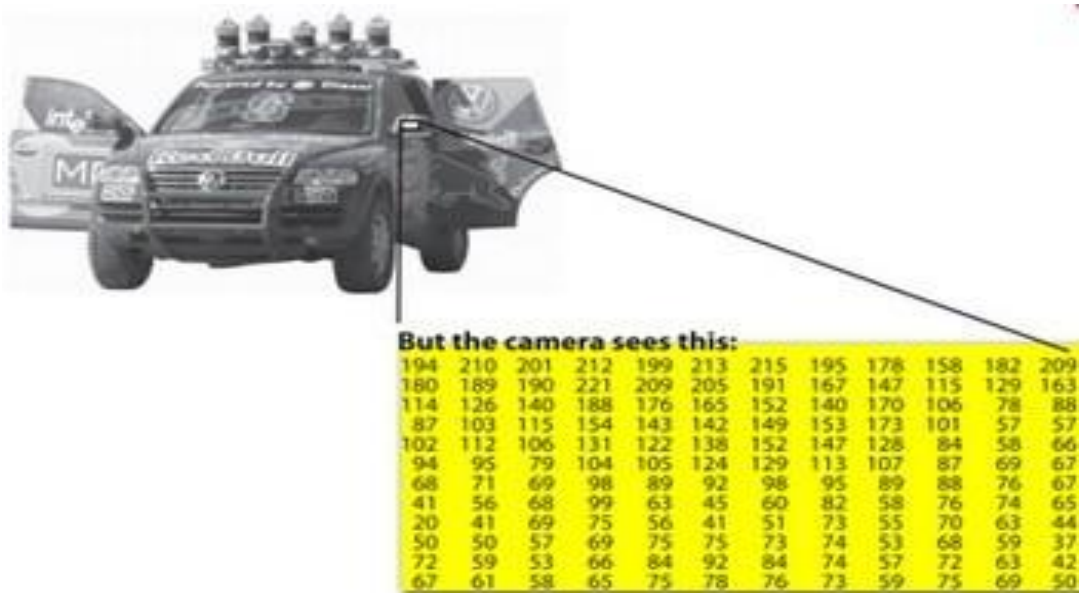

**Gambar 2. 1** Diagram hubungan computer vision dengan bidang lain (http://www.kdnuggets.com/2016/08/seven-steps-understanding-computer-vision.html)

## 1.2. *Computer Vision*

Computer Vision merupakan bidang antar berbagai disiplin ilmu pengetahuan yang berurusan tentang bagaimana komputer dapat memahami objek dari gambar ataupun video digital. Dalam perspektif engineering, bidang ini diusahakan untuk melakukan tugas secara otomatis, dan bekerja sebagaimana mata manusia dapat bekerja. **[3]**

Tugas Computer Vision meliputi kegiatan mendapatkan, mengolah, menganalisa, memahami citra digital, dan secara umum berurusan dengan ekstraksi data dari dunia nyata agar menghasilkan informasi numeris. **[4]**

Pada sistem pengolahan citra, komputer hanya menerima sederetan nilai dari kamera ataupun perangkat penyimpan (disk). Gambar 2.2 berikut menampilkan suatu gambar mobil. Pada gambar tersebut, objek yang kita lihat ialah sisi kaca spion. Namun apa yang komputer lihat hanyalah sederetan nilai. Tugas seorang pelaku Computer Vision ialah dengan mengolah nilai-nilai tersebut, sehingga dapat diidentifikasi bahwa objek tersebut merupakan spion. [5]




**Gambar 2. 2** Ilustrasi bagaimana komputer melihat
(Bradski p. 3, 2008)

## 1.3. OpenCV

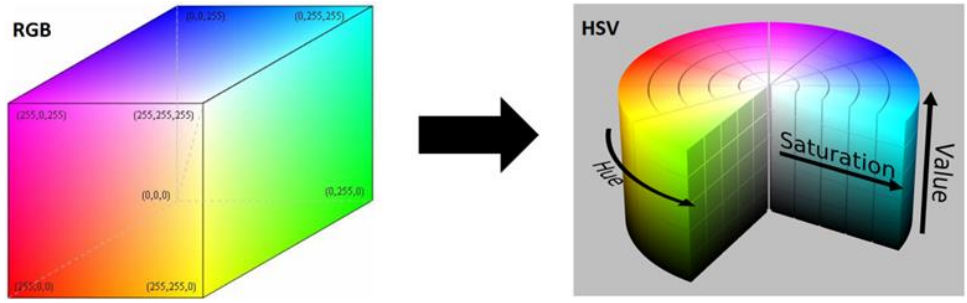
Untuk membantu aktivitas pemrograman dalam pengolahan citra maupun *computer vison*, terdapat *library* program yang mencakup hampir seluruh fungsi atau metode pemrograman *computer vision*, yaitu OpenCV. Dibuat sebagai *open source library*, pustaka ini boleh digunakan, didistribusikan, dan dikembangkan oleh siapa saja. *Library* ini awalnya ditulis dalam kode C dan C ++, namun juga terdapat pengembangan pada beberapa bahasa pemrograman lainnya, seperti Python, Ruby, Java, dan Matlab. *Library* ini dirancang untuk mengoptimalkan efisiensi komputasi dan

berfokus kuat pada aplikasi *real-time* [5]. Saat ini, *OpenCV library* dapat dijalankan di beberapa platform digital termasuk iOS, android, windows, Mac OS, Linux dan digunakan di berbagai mesin seperti telepon genggam, laptop, tablet, dan komputer pribadi.

## 1.4. Model Warna HSV (Hue, Saturation, Value)

HSV adalah sistem koordinat-silinder yang paling umum merepresentasikan poin dalam model warna RGB, yang mengatur ulang geometri RGB dalam upaya untuk perseptual yang lebih relevan daripada representasi koordinat kartesian [6].

Untuk memudahkan proses pengolahan citra, umumnya gambar yang dimodelkan dalam bentuk RGB terlebih dahulu diubah ke model HSV. Model warna HSV digunakan karena model ini menggambarkan warna yang sama dengan intuisi manusia untuk melihat warna. Seperti ditunjukkan pada Gambar 2.3 yang mana RGB mendefinisikan warna sebagai kombinasi warna merah, hijau, dan biru, HSV mendeskripsikan warna dengan menggunakan perbandingan yang lebih akrab. Hue mewakili jenis warna, Saturation mewakili bagaimana jenuh atau pudarnya warna tersebut, dan Value mewakili tingkat penerangan.




**Gambar 2. 3** Perbandingan model warna RGB dengan HSV
(https://people.eecs.berkeley.edu/~sequin/CS184/TOPICS/ColorSpaces/Color_0.html; https://en.wikipedia.org/wiki/HSL_and_HSV)

Untuk merubah model warna RGB ke HSV, Travis dalam [7] telah menyediakan formula berikut:

$$M = \max(R, G, B) \tag{1}$$

$$m = \min(R, G, B) \tag{2}$$

$$C = M - m \tag{3}$$

$$H' = \begin{cases} undefined, & if\ C = 0 \\ \frac{G-B}{C}\ mod\ 6, & if\ M = R \\ \frac{B-R}{C} + 2, & if\ M = G \\ \frac{R-G}{C} + 4, & if\ M = B \end{cases} \tag{4}$$

$$H = 60° \text{ x } H' \tag{5}$$

$$S = \begin{cases} 0, & if\ V = 0 \\ \frac{C}{V}, & otherwise \end{cases} \quad (6)$$

$$V = M \quad (7)$$

### 1.5. Pencocokan Kurva (*Curve Fitting)*

Pencocokan kurva adalah proses membangun kurva, atau fungsi matematis, yang paling sesuai dengan serangkaian titik data. Pencocokan kurva dapat melibatkan interpolasi, di mana kecocokan tepat dengan data diperlukan, atau smoothing, di mana fungsi untuk menghaluskan kurva dibuat sehingga dapat menyesuaikan data. Topik yang berkenaan dengan pencocokan kurva ialah adalah analisis regresi, yang lebih berfokus pada pertanyaan tentang inferensi statistik seperti seberapa banyak ketidakpastian hadir dalam kurva yang sesuai dengan data yang diamati dengan kesalahan acak. Pencocokan kurva dapat digunakan sebagai bantuan untuk visualisasi data, untuk mendapatkan nilai menggunakan fungsi untuk data yang tidak tersedia, dan untuk meringkas hubungan antara dua atau lebih variabel. Ekstrapolasi mengacu pada penggunaan kurva yang dipasang di luar jangkauan data yang diamati, dan dapat mencerminkan metode yang digunakan untuk membangun kurva sebesar data yang diamati. [8]

### 1.6. Interpolasi Polynomial

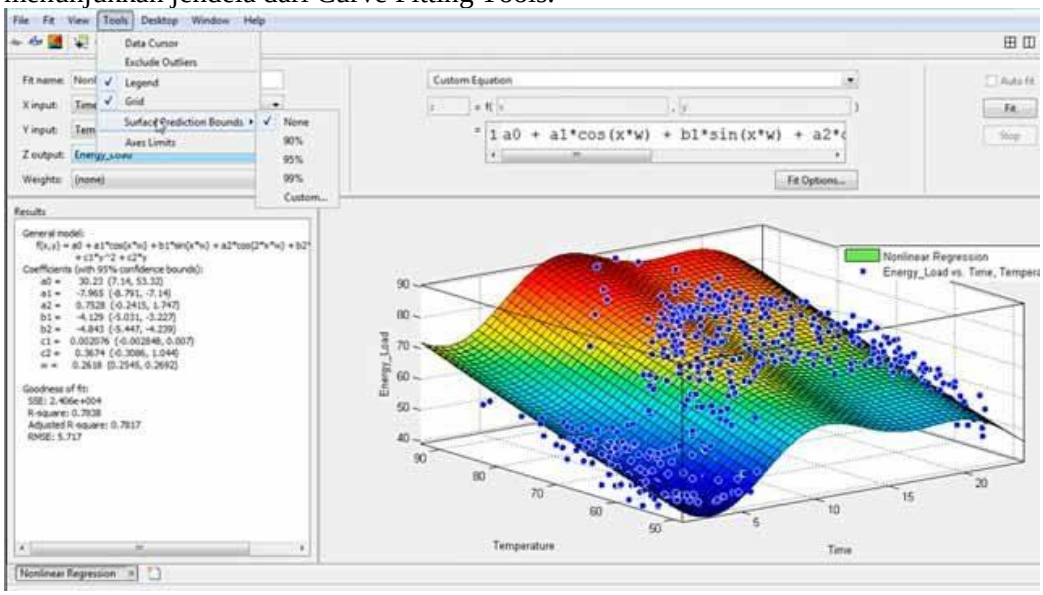
Interpolasi polynomial merupakan salah satu metode pencocokan kurva yang menginterpolasi data yang telah tersedia, baik oleh pengukuran, penyederhanaan polynomial, ataupun table yang telah tersedia di literatur. Interpolasi ini merupakan generalisasi dari interpolasi linear, yang mana fungsi diganti oleh polinomial dengan derajat lebih tinggi. Contoh dari fungsi polinomial 6 derajat ialag $f(x) = -0.0001521^6 – 0.003130x^5 + 0.007321x^4 - 0.3577x^3 + 0.225x^2 + 0.9038x$. [9] Formula untuk melakukan interpolasi polinomial dapat menggunakan formula polynomial Lagrange Eq. (8):

$$p(x) = \sum_{i=0}^{n} \left( \prod_{\substack{0 \le j \le n \\ j \ne 1}} \frac{x - xj}{xi - xj} \right) yi \quad (8)$$

### 1.7. Matlab Curve Fitting Toolbox

Curve Fitting Toolbox merupakan aplikasi dan fungsi untuk melakukan pencocokan kurva dan surface pada data. Toolbox ini memungkinkan untuk melakukan analisis data eksploratori, data preprocess dan post-process, membandingkan model kandidat, dan menghapus outlier. Dapat digunakan untuk melakukan analisis regresi dengan menggunakan *library* model linier dan nonlinier yang disediakan atau yang ditentukan oleh pengguna sendiri. *Library* menyediakan parameter pemecah yang dioptimalkan dan kondisi awal untuk meningkatkan kualitas kurva. Toolbox ini juga mendukung teknik pemodelan nonparametrik, seperti splines, interpolation, dan smoothing. [10]

Setelah membuat pencocokan kurva, langkah selanjutnya dapat diterapkan berbagai metode pasca pengolahan untuk perencanaan, interpolasi, dan ekstrapolasi. Dan, memperkirakan interval, serta menghitung integral dan derivatif. Gambar 2.4 menunjukkan jendela dari Curve Fitting Tools.




**Gambar 2. 4** Perbandingan model warna RGB dengan HSV
(https://www.mathworks.com/products/curvefitting.html)

## 1.8. Segmentasi Obyek Citra

Terdapat berbagai macam cara untuk mensegmentasi obyek dalam pengolahan citra. Obyek yang di segmentasi biasanya akan dihasilkan dalam bentuk gambar biner. Cara paling gampang melakukan segmentasi ialah dengan melakukan thresholding [11]. Thresholding bekerja dengan memilih pixel dengan membatasi intensitas pixel mana saja yang masuk kategori obyek yang ingin didapat. Berikut fungsi matematis dari proses ini.

$$dst_H = \begin{cases} 1, & if\ vLwr(x,y)_H \le src(x,y)_H \le vUpr(x,y)_H \\ 0, & otherwise \end{cases} \quad (9)$$

$$dst_S = \begin{cases} 1, & if\ vLwr(x,y)_S \le src(x,y)_S \le vUpr(x,y)_S \\ 0, & otherwise \end{cases} \quad (10)$$

$$dst_V = \begin{cases} 1, & if\ vLwr(x,y)_V \le src(x,y)_V \le vUpr(x,y)_V \\ 0, & otherwise \end{cases} \quad (11)$$

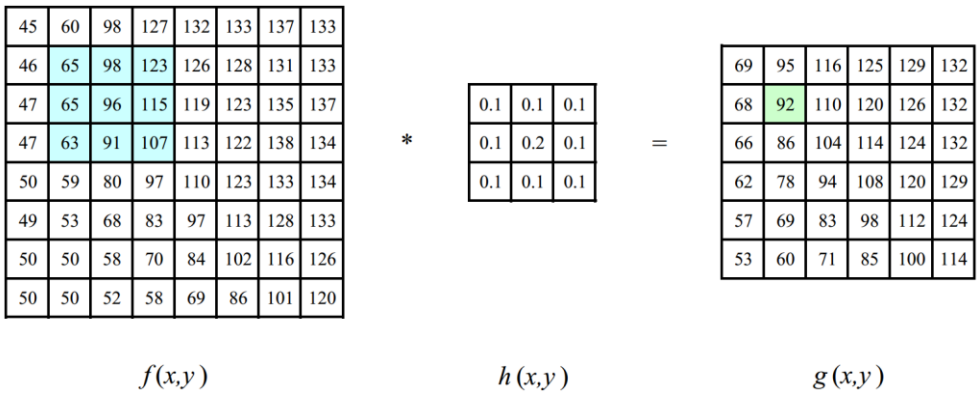
$$dst(x,y) = \begin{cases} 255, & if\ \ dst_H = 1 \wedge dst_S = 1 \wedge dst_V = 1 \\ 0, & otherwise \end{cases} \quad (12)$$

Setiap pixel dalam gambar RGB, maupun HSV merupakan vector. Jika setiap pixel dalam keseluruhan matrix merupakan vector itu berarti pixel tersebut menampung lebih dari satu nilai, pada HSV ada 3 yaitu untuk Hue, Saturation, dan Value. Karena gambar output merupakan array bertipe *unsigned integer* maka nilai

teringginya ialah 255 (semua bit berlogika 1) direpresentasikan dengan warna putih dan terendahnya ialah 0 (semua bit berlogika 0) di representasikan dengan warna hitam. Jadi, melalui persamaan ini, kordinat pixel yang memiliki nilai intensitas yang berada di dalam range yang telah ditentukan akan berwarna putih, dan yang lainnya akan berwarna hitam.

## 1.9. Pengurangan Tingkat Noise

Ada beberapa cara untuk menghilangkan piksel yang tidak diinginkan dari gambar. Cara sederhana untuk melakukan tugas ini adalah melakukan smoothing. Jenis operator yang paling umum digunakan adalah filter linier, dimana nilai piksel keluaran ditentukan sebagai jumlah nilai masukan piksel tertimbang. Seperti yang ditunjukkan pada Gambar 2.5, gambar di sebelah kiri melingkupi filter di tengah untuk menghasilkan gambar di sebelah kanan. Pixel biru muda menunjukkan filter sumber untuk keluaran pixel hijau muda [12]. Filter yang digunakan dapat disusun berdasarkan model statistic, seperti Gaussian Filter, Median Filter, Mean Filter, dll.




**Gambar 2. 5** Convolution
(Szeliski p. 112, 2010)

## 1.10. Crane

Crane merupakan salah satu jenis mesin, umumnya dilengkapi dengan tali kerekan, tali kawat atau rantai, dan katrol, yang dapat digunakan baik untuk mengangkat dan menurunkan benda dan memindahkannya secara horisontal. Peralatan ini digunakan untuk mengangkat benda-benda berat dan mengangkutnya ke tempat lain. Perangkat menggunakan satu atau lebih mesin sederhana untuk menciptakan keuntungan mekanis dan dengan demikian memindahkan beban di luar kemampuan normal manusia. Crane biasanya digunakan di industri transportasi untuk bongkar muat barang, di industri konstruksi untuk memindahkan material, dan di industri manufaktur untuk perakitan alat berat. [13]

*Crane* menjadi peralatan penting di dunia industri selain dari konveyor dan truk. Tidak hanya industri, *crane* juga digunakan pada kontruksi dan bidang perkapalan. Fungsi utama sebuah *crane* adalah untuk mengangkat benda tertentu dan memindahkannya ke tempat yang diinginkan. Terdapat tiga gerakan umum pada *crane*,

yaitu *hoisting*, *derricking*, dan *slewing*. Umumnya, *crane* digerakkan oleh motor listrik, mesin diesel, atau gabungan keduanya. *Crane* yang digunakan pada berbagai industri, dapat diklasifikasikan sebagai berikut:

a. Berdasarkan konstruksi:

- *Rotary crane*, misalnya *tower crane* dan *jib crane* dapat dilihat pada Gambar 2.6.

**Gambar 2. 6** Jib crane
**(**http://www.trademark-hoist.com**)**

- *Mobile crane*, misalnya *rail mounted* dan *floating crane* dapat dilihat pada Gambar 2.7. yang diambil dari situs.

**Gambar 2. 7** Floating crane
**(**http://www.shipseller.net**)**

- *Bridge crane*, misalnya *overhead travelling crane* dan *gantry crane* dapat dilihat pada Gambar 2.8.


**Gambar 2. 8** Gantry crane
(https://www.shuttlelift.com)

b. Berdasarkan kegunaan:
   - *Indoor crane*
   - *Outdoor crane*

c. Berdasarkan beban yang diangkat:
   - *Light duty*
   - *Medium duty*
   - *Heavy duty*

d. Berdasarkan perpindahan:
   - *Stationary crane*
   - *Mobile crane*

Pada setiap *crane* terdapat bagian-bagian penting yang menunjang fungsi dari *crane* tersebut. Berikut bagian-bagian penting dan fungsi dari *crane* pada umumnya:

a. Struktur penunjang
   Struktur penunjang umumnya berupa dinding atau *girder*. Dinding terbuat dari semen, sedangkan *girder* terbuat dari berbagai bagian atau sudut. *Girder* biasanya menunjang *hoisting trolley*, di mana terdapat beban statis dan dinamis.

b. *Hoisting mechanism*
   *Hoisting mechanism* merupakan mekanisme inti dari sebuah *crane*. Biasanya dioperasikan secara manual atau menggunakan penggerak. Mekanisme ini terdiri dari elemen *hoisting*, seperti *wire rope* atau *chain*, motor penggerak, *gear*, katrol, *sprocket*, *drums*, dan lain-lain.

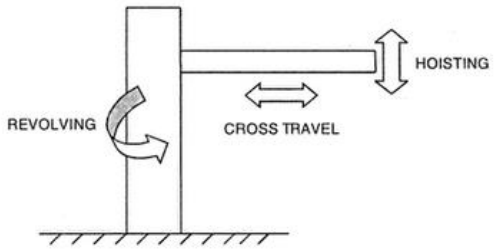
c. *Load lifting attachments*
*Load lifting attachments* merupakan peralatan tambahan untuk mengangkat beban. Biasanya menggunakan *hook*. Peralatan ini dibedakan berdasarkan beban yang akan diangkat.

d. Peralatan kontrol
Peralatan kontrol pada *crane t*erdiri dari peralatan untuk *safe working* dari sebuah *crane*. Terdiri dari *brakes*, *limit switches*, *arresting gears*, *relay*, dan sensor.

e. Penggerak
Penggerak pada *crane* dapat berupa elektrikal, *gear drive*, atau diesel generator.

f. *Crab* atau *cabin*
*Cabin* merupakan tempat operator mengoperasikan *crane*. Letaknya sangat penting dalam operasional *crane*.

Selain bagian-bagian penting, terdapat juga gerakan dasar yang dapat membuat sebuah beban diangkat oleh *crane* seperti yang diilustrasikan pada Gambar 2.9. [14]




**Gambar 2. 9** Gerakan dasar *crane*
(Arora p. 40, 2007)

a. *Hoisting motion*: merupakan gerakan yang khusus berfungsi untuk mengangkat dan menurunkan beban.
b. *Radial movement of load*: pada gerakan ini beban berpindah terhadap *fixed frame* atau *centre of crane*.
c. *Revolving motion*: gerakan ini memungkinkan beban berputar di sumbu vertical sebesar 360°.
d. *Travelling the crane with load*: gerakan ini memungkinkan seluruh *crane* berpindah bersama beban.

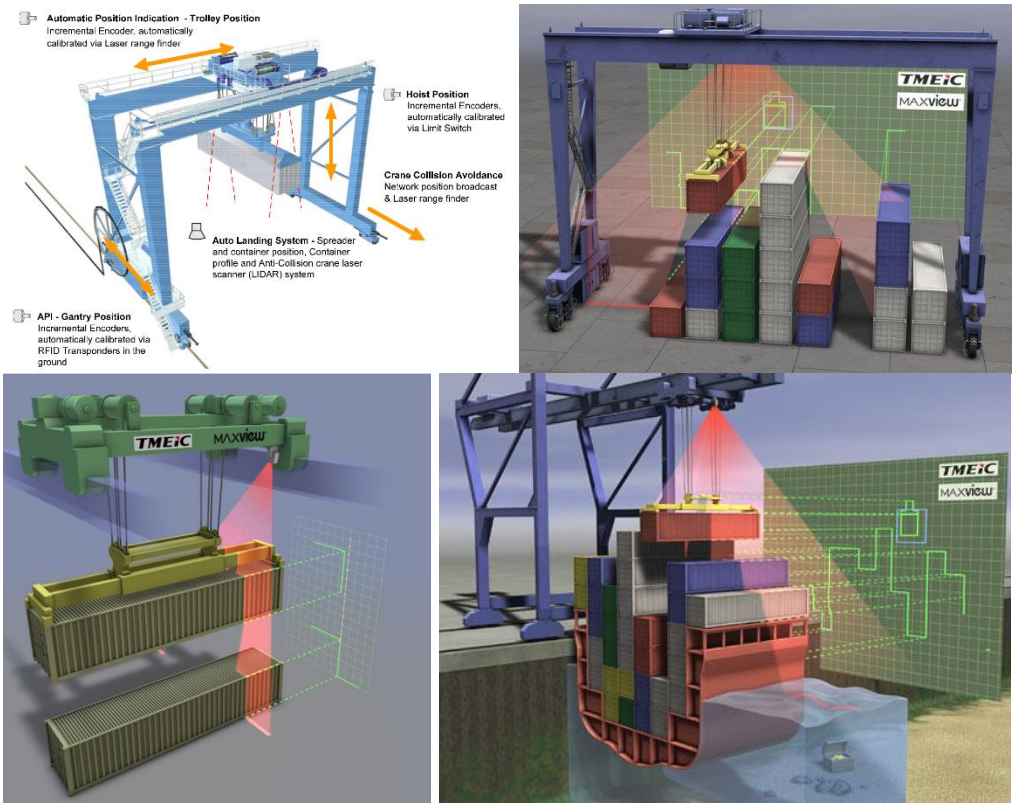
## 1.11. Automatic Stacking Crane

Proses pengelolaan peti kemas atau kontainer pada lapangan penumpukan (*stacking yard*) membutuhkan peralatan yang sesuai. Pada *stacking yard* yang berukuran kecil, proses penumpukan biasanya dilakukan menggunakan *forklift* atau *reach stacker*. Pada *stacking yard* yang lebih besar, proses penumpukan biasanya dibantu oleh *Rail Mounted Gantry* ataupun *Rubber Tyred Gantry*.

Saat ini proses penumpukan pada *stacking yard* di beberapa tempat telah mengalami otomatisasi. *Rail Mounted Gantry* yang sebelumnya digunakan secara manual, yaitu dilakukan dengan pengambilan keputusan oleh manusia, kini telah berganti menjadi otomatis, yang mana proses pengambilan keputusan dilakukan oleh mesin. *Rail Mounted Gantry* yang mampu bekerja secara otomatis ini disebut dengan

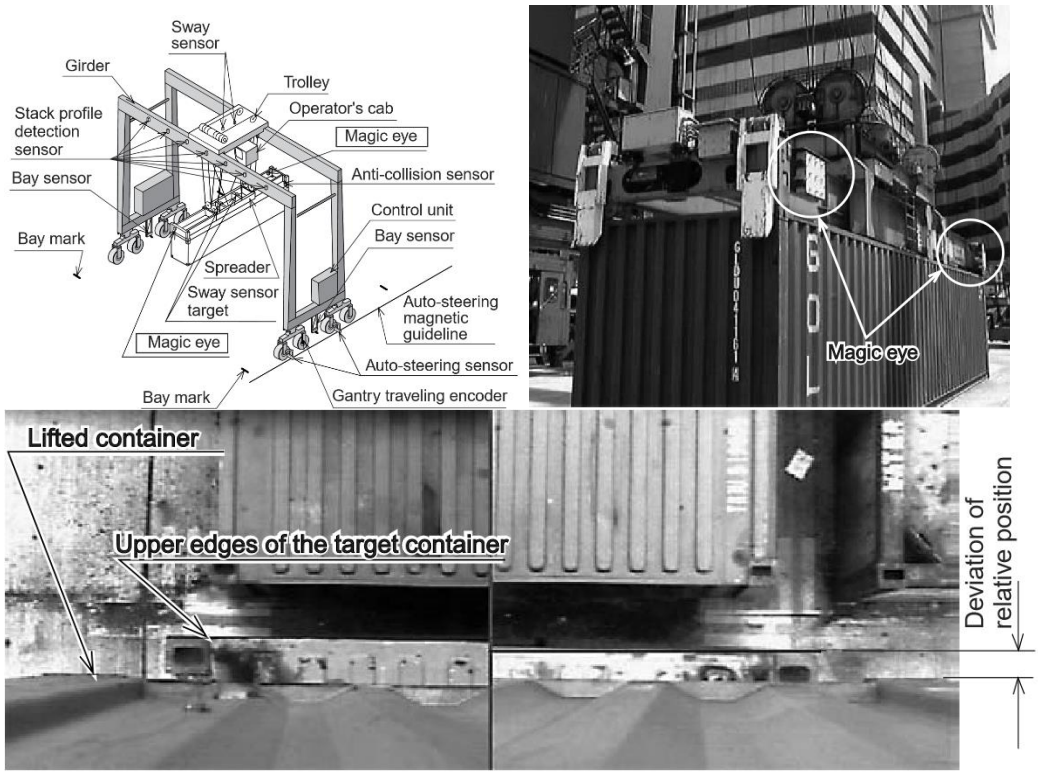
*Automated Rail Mounted Gantry* atau untuk yang lebih umum bisa disebut *Automatic Stacking Crane*.

*Automatic Stacking Crane* dapat bekerja dengan berbagai cara untuk melakukan tugasnya. Sebagai contoh ASC (*Automatic Stacking Crane*) yang dibuat oleh TMEIC Maxview. TMEIC maxview menggunakan sensor laser yang diletakkan di *trolley* untuk mendeteksi gerakan *spreader* dan mendeteksi posisi terhadap kontainer sekitar, ilustrasi dapat dilihat pada Gambar 2.10. [14]




**Gambar 2. 10** Ilustrasi sistem otomatis pada TMEIC Maxview Crane
(TMEIC Corporation, 2017)

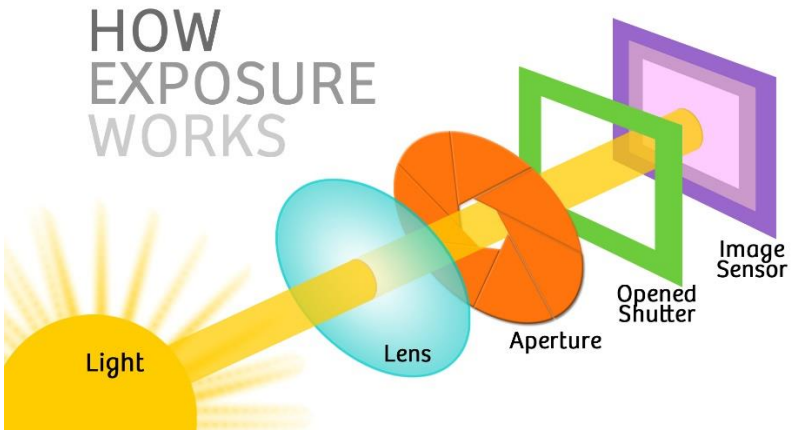
Pada crane Mitsubishi Heavy Industries untuk melakukan proses *positioning* ASC menggunakan cara yang berbeda. MHI menggunakan kamera yang diletakkan pada *spreader* untuk mendeteksi deviasi antara kontainer yang dibawa dengan kontainer dibawahnnya. Kamera ini dinamakan magic-eye oleh MHI, terdapat 4 buah dan diletakkan pada setiap sudut *spreader*, ilustrasi dapat dilihat pada Gambar 3.10. [15]

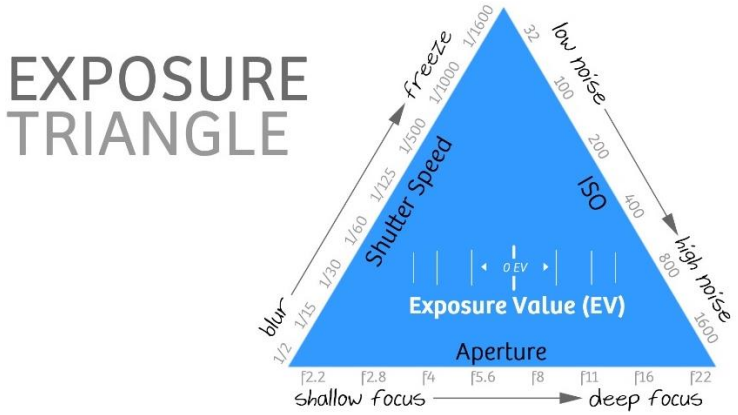



**Gambar 2. 11** Iustrasi sistem otomatis pada MHI Automated Transfer Crane (Obata p. 1-3, 2003)

## 1.12. *Exposure*

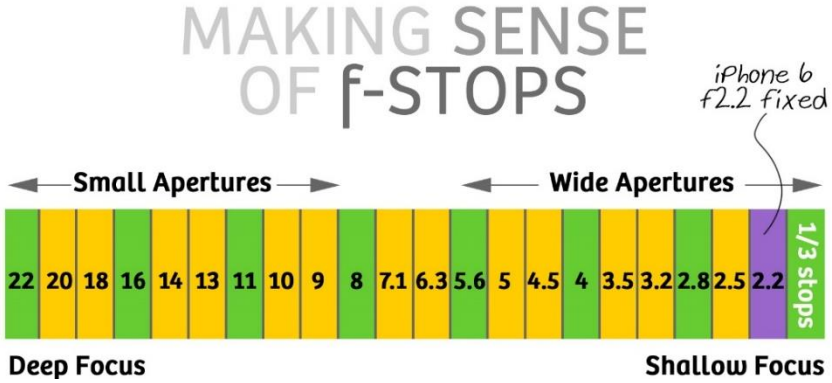
*Exposure* merupakan proses penerimaan cahaya melalui lensa kamera ke sensor digital. Diilustrasikan oleh Gambar 2.12, Setiap *exposure* dipengaruhi oleh 3 faktor, yaitu *aperture*, *shutter speed*, dan ISO.





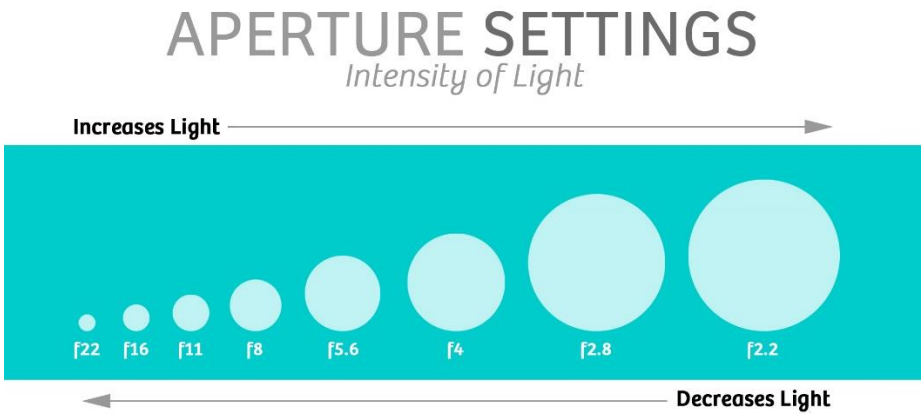

**Gambar 2. 12** Iustrasi cara kerja *exposure*
(http://snapsnapsnap.photos/a-beginners-guide-for-manual-controls-in-iphone-photography-shutter-speed/)

*Apertur* adalah bukaan (secara teknis disebut 'iris') pada lensa. *Aperture* mengacu kepada intensitas cahaya yang mana diukur dalam 'f-stops'. Skala 'f-stop' terlihat seperti Gambar 2.13 (pada 1/3 stops):




**Gambar 2. 13** Skala f-stops
(http://snapsnapsnap.photos/a-beginners-guide-for-manual-controls-in-iphone-photography-shutter-speed/)

f-stop yang semakin kecil berarti *aperture* semakin besar, dan sebaliknya f-stop yang semakin besar berarti *aperture* semakin kecil. Sebagaimana contoh di Gambar 2.14, f2.2 lebih besar daripada f22.

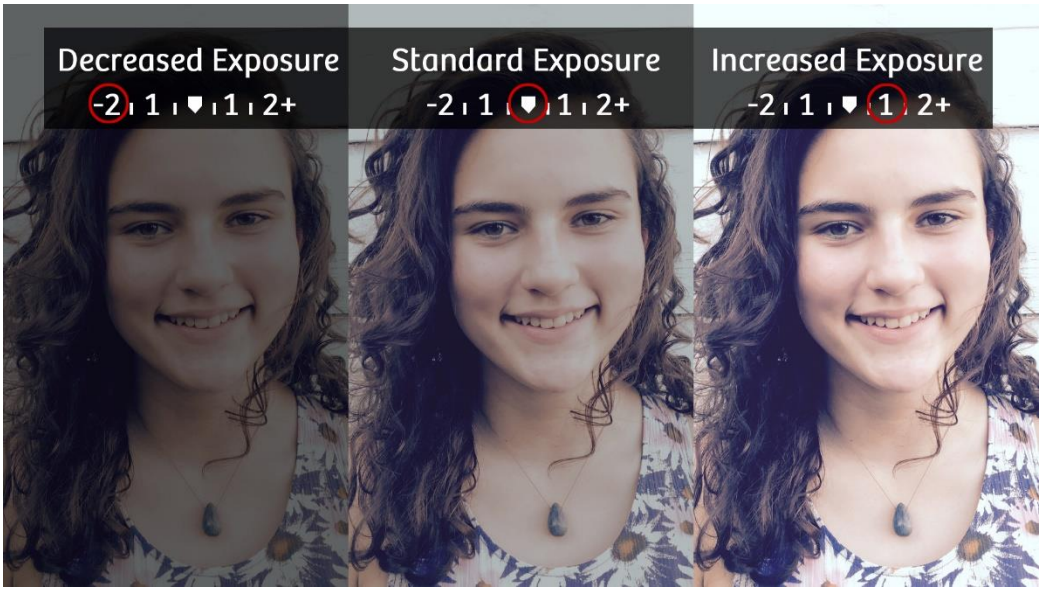



**Gambar 2. 14** Pengaturan *aperture*
(http://snapsnapsnap.photos/a-beginners-guide-for-manual-controls-in-iphone-photography-shutter-speed/)

Jika *aperture* merupakan intensitas cahaya yang sampai ke sensor kamera, *shutter speed* merupakan durasi cahaya yang sampai ke sensor. *Shutter speed* yang semakin lambat (seperti ½ detik, ¼ detik, 1/8 detik, 1/15 detik) menghasilkan pergerakan 'blur', karena cenderung menghasilkan gambar yang blur dan seolah-olah tertarik oleh pergerakan obyek. Sedangkan, *shutter speed* yang lebih cepat (seperti 1/250 detik, 1/500 detik, 1/1000 detik) cenderung menghasilkan obyek yang diam, atau disebut '*freeze*'.

ISO mengacu kepada keseluruhan sensitivitas sensor terhadap cahaya. Nilai ISO yang yang semakin rendah menghasilkan citra yang lebih gelap dan noise atau gangguan yang ditimbulkan juga semakin rendah. Sedangakan, ISO yang semakin besar mengasilkan citra yang lebih terang dan menimbulkan noise atau gangguan yang semakin besar pula.

*Exposure Value* merupakan angka yang digunakan untuk merepresentasikan kombinasi *exposure* (dari *aperture*, *shutter speed*, dan ISO) yang sesuai dan dapat diterapkan. Setiap level EV (*Exposure Value*) sama terhadap satu satuan pengaturan dari *aperture*, *shutter speed*, dan ISO. Seperti ditunjukkan dalam Gambar 2.15 menaikkan *Exposure Value* membuat citra menjadi lebih terang, dan menurunkan *Exposure Value* membuat citra menjadi lebih gelap. [16]




**Gambar 2. 15** Ilustrasi pengaturan pada *exposure value*
(http://snapsnapsnap.photos/a-beginners-guide-for-manual-controls-in-iphone-photography-shutter-speed/)

## 1.13. Fungsi Eksponensial

Dalam matematika, fungsi eksponensial merupakan fungsi dalam bentuk $f(x)=b^x$, yang mana variabel masukan $x$ berfungsi sebagai eksponen. Fungsi dalam bentuk $f(x)=b^{x+c}$, dimana c adalah konstanta, juga dianggap sebagai fungsi eksponensial dan dapat ditulis ulang sebagai $f(x)=ab^x$, dengan $a=b^c$.

Sebagai fungsi dari variabel bilangan real, fungsi eksponensial dicirikan berdasarkan fakta bahwa tingkat pertumbuhan dari turunannya secara langsung sebanding terhadap nilai fungsinya. Konstanta proporsionalitas dari hubungan ini merupakan logaritma natural dari basis $b$, yaitu

$$\frac{d}{dx}(b^x) = b^x \boldsymbol{log_e(b)} \tag{13}$$

Konstanta e≈2.71828… merupakan basis unik yang mana konstanta proporsionalitas akan bernilai 1, jadi fungsi turunannya adalah:

$$\frac{d}{dx}(e^x) = e^x log_e(e) = e^x \tag{14}$$

Karena merubah basis dari fungsi eksponensial semata-mata hanya akan merubah penampilan dari faktor konstanta, secara komputasional hal ini cukup sesuai untuk membatasi pelajaran fungsi eksponensial pada matematika analisis ke dalam pelajaran fungsi khusus yang secara konvensional disebut "fungsi eksponensial natural", atau secara sederhana "fungsi eksponensial" saja, ditunjukkan oleh

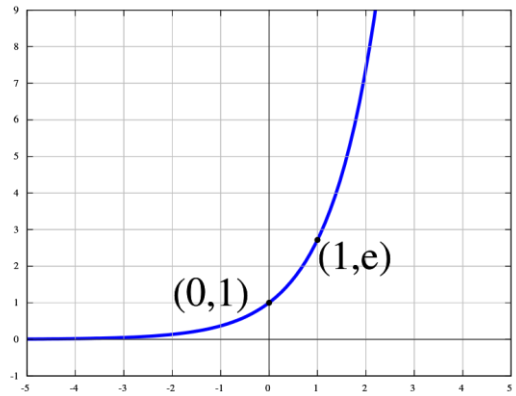
$$x \to e^x atau \exp(\cdot) \tag{15}$$

Fungsi eksponensial memenuhi asas identitas perkalian

$$e^{x+y} = e^x e^y, untuk\ semua\ \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R} \tag{16}$$

Pada faktanya identitas perkalian ini bisa menjadi bilangan eksponen yang kompleks. Hal ini ditunjukkan dengan setiap solusi kontinu, dan bukan nol dari persamaan fungsi $f(x+y)=f(x)f(y)$ merupakan fungsi eksponensial, f: $\mathbb{R} \to \mathbb{R}, x \longmapsto b^x$, dengan b > 0. Argumen fungsi eksponensial dapat berupa bilangan real ataupun bilangan kompleks ataupun malah jenis lain dari obyek matematik seperti matrix misalnya.

Karena fungsi ini sering ditemukan dimana-mana, baik dalam matematika murni maupun terapan, matematikawan W. Rudin berpendapat bahwa fungsi eksponensial merupakan fungsi paling penting dalam matematika. Pada hal matematika terapan, fungsi eksponensial memodelkan hubungan yang mana konstanta berubah dalam variabel independen memberikan perubahan proporsional yang sama terhadap variabel dependen. Seperti halnya yang terjadi pada ilmu sosial maupun pengetahuan alam, ditemukan dalam konteks ilmu fisika, kimia, teknik, biologi matematis, dan ekonomi.




**Gambar 2. 16** Fungsi eksponensial natural
(Peter John Acklam - Own work, CC BY-SA 3.0)

Seperti terlihat dalam Gambar 2. 16, grafik $y = e^x$ merupakan gradien yang bergerak menaik, dan meningkat lebih cepat sebagaimana *x* meningkat. Grafik selalu berada diatas sumbu-*x*, namun dapat sangat dekat dengan sumbu-*x*, oleh karena itu sumbu-*x* merupakan asimtot horisontal. Gradien dari setiap garis singgung pada tiap titik sama dengan koordinat-*y* pada titik tersebut, seperti yang ditunjukkan dalam fungsi turunannya. Fungsi inverse dari fungsi ini ialah fungsi logaritma natural, ditunjukkan dengan **log, ln,** ataupun $\log_e$. [18]

## 1.14. Android SDK (Software Development Kit)

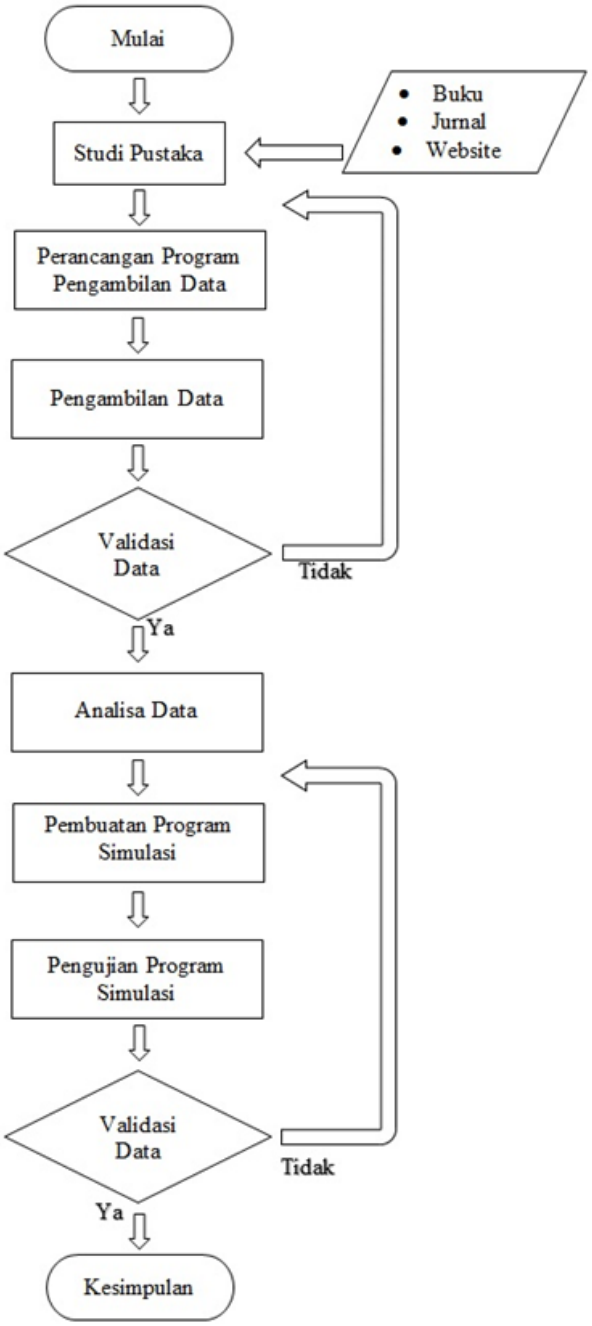
Android Software Development Kit merupakan kumpulan *tools* yang komprehensif untuk mengembangkan program android. Paket *tools* ini termasik *debugger*, *library*, emulator, dokumentasi, contoh kode, tutorial. SDK ini adapat digunakan pada berbagai platform, yaitu Linux, MAC OS, maupun Windows.

Sampai sekitar akhir 2014, IDE (*Integrated Development Environment*) yang didukung secara ofisial oleh Android adalah ‘Eclipse’. Eclipse dapat digunakan untuk mengembangkan android dengan menggunakan plugin ADT (Android Development Tools). Terdapat IDE lain juga yang dapat digunakan, yaitu ‘Intellij IDEA’ dan ‘NetBeans’. Di tahun 2015, Eclipse bukan lagi IDE ofisial untuk mengembangkan android. Namun yang menjadi IDE ofisial adalah ‘Android Studio’, dibuat oleh Google,

dipersembahkan oleh Intellij. Akan tetapi pengembang atau *developer* bebas untuk menggunakan IDE manapun, tapi Google menjelaskan bahwa ADT secara ofisial tidak berlaku lagi sejak akhir 2015 untuk berfokus pada Android Studio sebagai IDE ofisial. [19]

Pada Tugas Akhir ini penulis menggunakan IDE dari Eclipse dan library dari OpenCV khusus untuk Android untuk mengembangkan program yang diterapkan pada perangkat android.

# BAB III
# METODOLOGI




Gambar 3. 1 Flowchart metodologi penelitian

Analisa teoritis dan sistematis terhadap metode yang digunakan merupakan hal yang penting. Kegiatan ini dan pelaporannya mampu menjadi indikator kevalidan suatu penelitian. Pada skripsi ini metode penelitian yang dilakukan ialah eksperimen/percobaan. Berikut langkah –langkah dalam penelitian yang akan dilakukan penulis:

### 3.1. Studi Pustaka

Studi pustaka berguna untuk memperkaya pengetahuan terkait penelitian yang akan dilakukan. Dari studi pustaka dapat diperoleh pula informasi mengenai masalah-masalah lain yang dapat timbul, dan berbagai pemecahan masalah yang telah dilakukan untuk mengatasi masalah yang sama. Dari pengetahuan-pengetahuan tersebut peneliti dapat merancang penelitian dengan dasar yang kuat.

### 3.2. Pembuatan Program Pengambilan Data

Dalam kegiatan ini penulis merancang program untuk mendapatkan nilai representasi intensitas cahaya baik dari Lightmeter kamera dan nilai warna dari beberapa objek warna serta berbagai proses lain di device android. Selain itu penulis mempersiapkan peralatan untuk pengkondisian eksperimen.

### 3.3. Pengambilan Data

Pada tahap ini penulis melakukan percobaan menggunakan program yang telah dirancang terhadap berbagai objek warna tetap, yaitu Merah, Hijau, Biru. Hal yang menjadi variable kontrol dalam percobaan ini ialah perlengkapan yang digunakan, dan posisi perlengkapan. Kondisi cahaya matahari dan nilai *exposure*e merupakan variabel bebas dalam penelitian ini. Variabel terikat yang merupakan konsekuensi dari variabel bebas ialah nilai warna dari objek warana acuan.

### 3.4. Analisa Data

Dari data yang telah didapat, dianalisa hubungan antara variabel bebas dan terikat. Dari data tersebut juga dapat dibuat diagram grafik-nya, dan dilihat tren data-nya. Dengan metode interpolasi dapat diketahui perkiraan nilai warna identifikasi berdasarkan intensitas cahaya-nya, dari hal ini pula dapat ditentukan formula yang menunjukkan kerja-nya.

### 3.5. Pembuatan Program Simulasi

Tahap ini dilakukan perancangan program dengan memasukkan formula yang telah ditentukan dari analisa sebelumnya. Pada hal ini yang dikontrol ialah nilai warna dari objek yang diacu. Selain itu, pada tahap ini program simulasi penerapan untuk mendeteksi gerakan-gerakan pada crane dapat dibuat.

### 3.6. Percobaan Program

Program yang telah dibuat sebelumnya diuji dan dianalisa kinerjanya. Jika masih diras kuran baik, maka dilakuka pemrograman ulang.

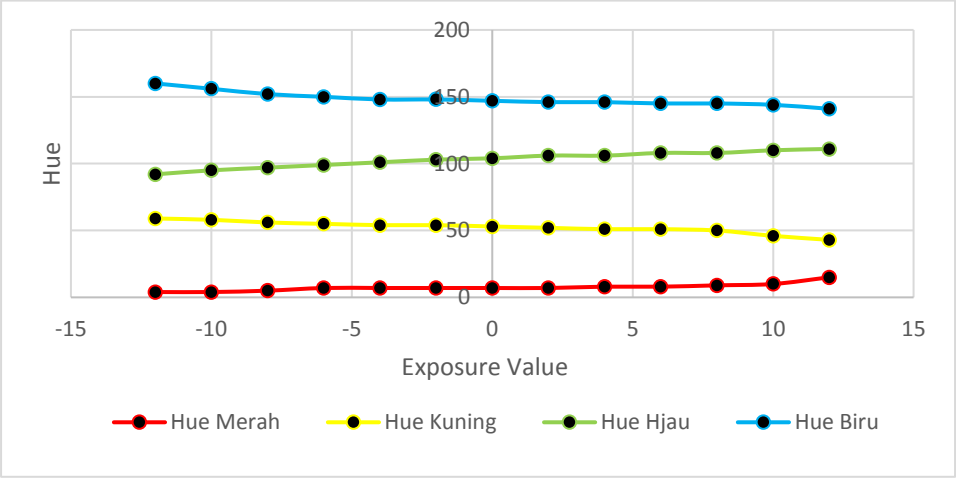
# BAB IV
# HASIL DAN PEMBAHASAN

## 4.1. Analisa Warna Model HSV Tehadap Perubahan Intensitas Cahaya

Pada sub-bab ini dibahas tentang data hasil pengambilan menggunakan perangkat android dengan program yang telah dibuat. Data yang diambil berupa hubungan antara properti masing-masing warna obyek (Hue, Saturation, Value) dengan nilai *exposure* maupun dengan intensitas cahaya.

### 4.1.1. Nilai *Exposure* – Hue

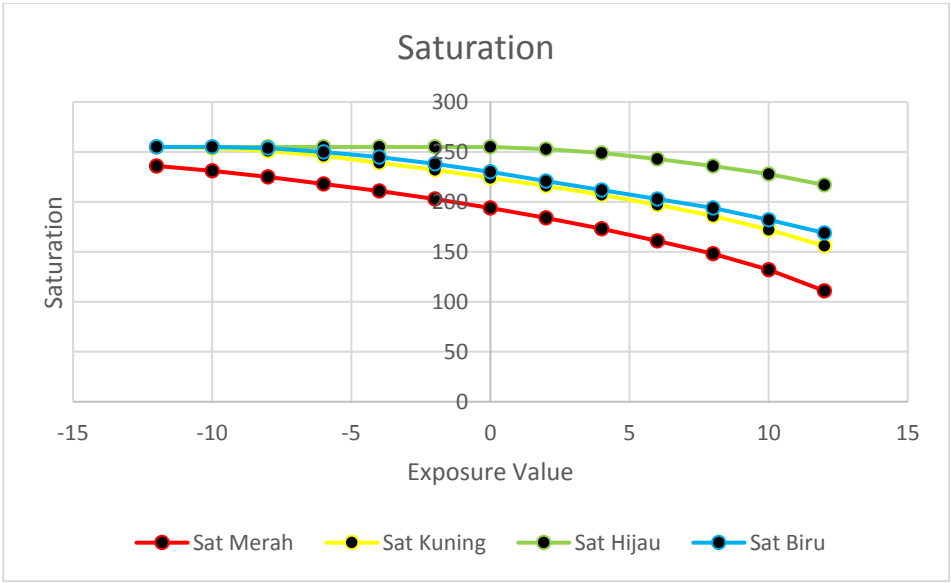
Pada Gambar 4.1 ditampilkan grafik hubungan antara nilai *exposure* pada perangkat android dengan nilai Hue (HSV) pada masing-masing obyek warna yang diamati. Pengambilan data dilakukan pada ruangan yang tertutup dengan kondisi cahaya sekitar yang cenderung stabil, dan dengan pengaturan *white balance* yang tetap, yaitu *daylight*. Terlihat dalam hubungan EV-Hue ini, perubahan pada *exposure* cenderung menyebabkan perubahan pada nilai Hue, namun tren yang terjadi berbeda-beda antar satu obyek warna dengan lainnya. Hal ini dapat disimpulkan, perubahan nilai Hue yang terjadi terhadap perubahan *exposure* perangkat bergantung dimana nilai Hue tersebut diacu.




Gambar 4. 1 Grafik respon Hue terhadap *exposure*

### 4.1.2. Nilai *Exposure* – Saturation

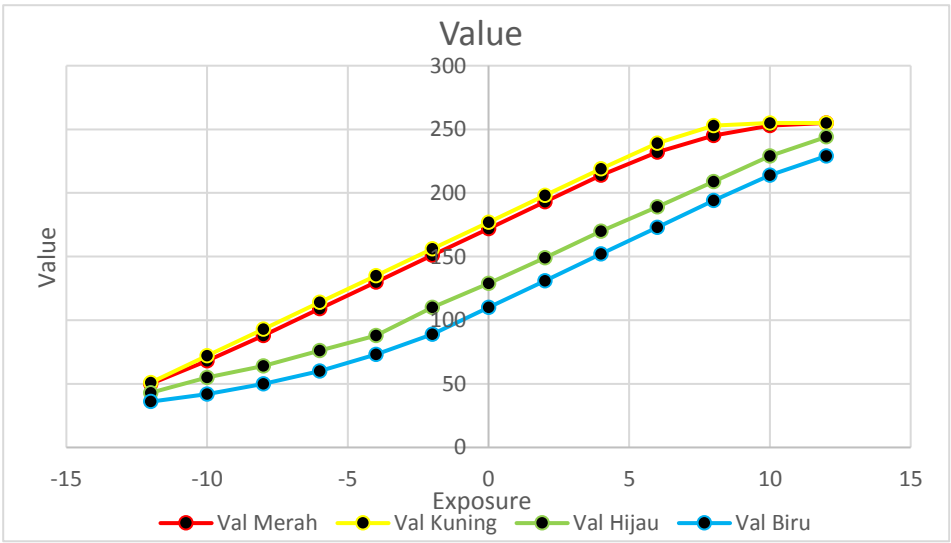
Pada Gambar 4.2 merupakan grafik yang menghubungkan nilai *exposure* pada perangakat android dengan nilai Saturation (HSV) dari masing-masing warna yang diamati. Nilai *exposure* yang tersedia pada android API (Application Program Interface) ialah berkisar antara -12 sampai 12, dimana semakin besar nilai *exposure* maka paparan terhadap cahaya juga semakin besar. Pada grafik EV-Saturation ini Nampak bahwa semakin besar nilai *exposure*e (Exposure Value) maka semakin kecil nilai Saturation (HSV) dari masing-masing obyek warna.




**Gambar 4. 2** Grafik respon Saturation terhadap *exposure*

### 4.1.3. Nilai *Exposure* – Value

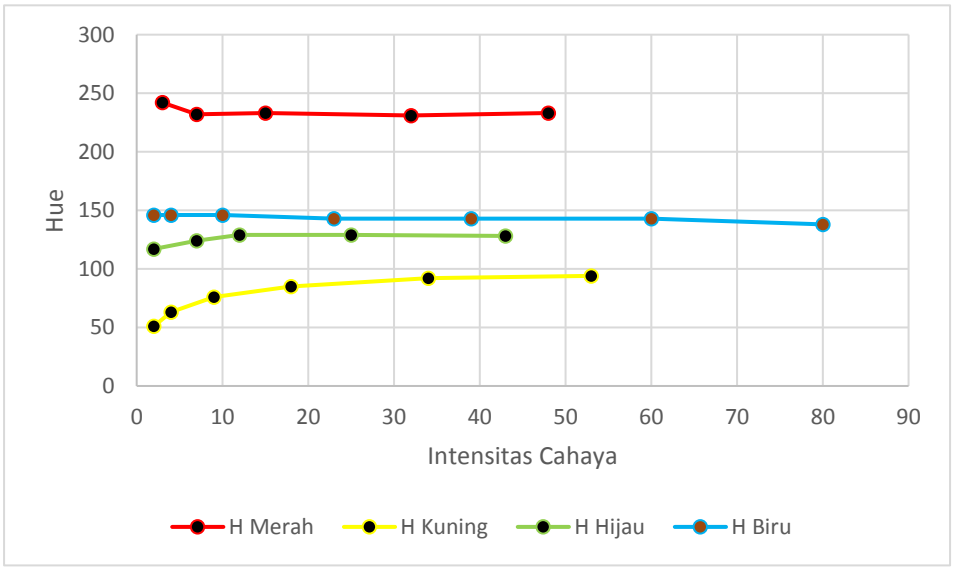
Jika pada hasil data sebelumnya yaitu respon hubungan antara nilai *exposure* dengan Saturation berbanding terbalik, lain halnya dengan respon hubungan antara nilai *exposure* dengan Value (HSV). Seperti terlihat pada Gambar 4.3 semakin besar nilai *exposure* maka semakin besar pula nilai Value (HSV). Kurva yang terbentuk pada grafik ini cenderung berbentuk sigmoid, dimana tingkat perubahan semakin kecil jika mendekati batas bawah (0) atau atas (255) jika tiap channel berukuran 8-bit.




**Gambar 4. 3** Grafik respon Value terhadap *exposure*

### 4.1.4. Intensitas Cahaya – Hue

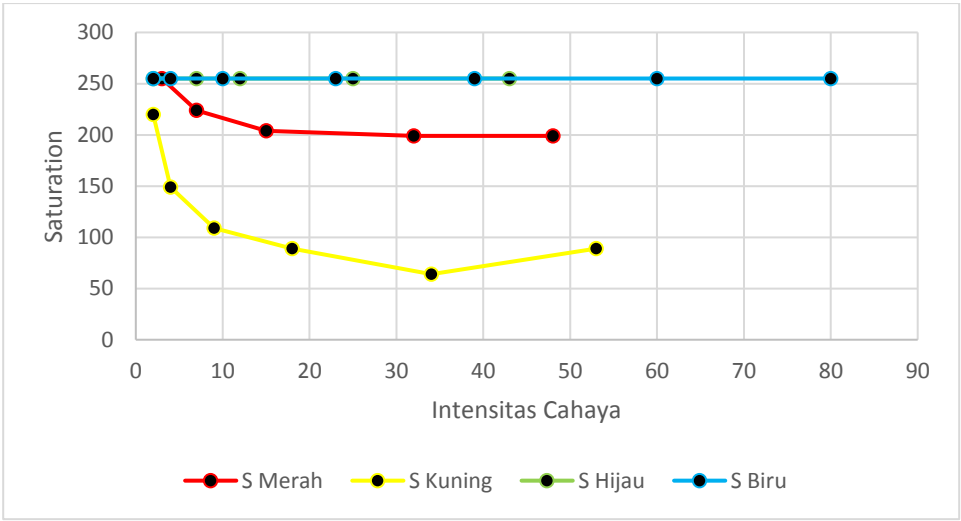
Pada gambar grafik sebelumnya telah disajikan mengenai hubungan antara masing-masing properti warna HSV dengan *exposure* pada perangkat yang mana juga merepresentasikan tingkat penerimaan cahaya. Pada grafik kali ini akan disajikan hubungan intensitas cahaya yang diukur menggunakan sensor cahaya yang memanfaatkan LDR (*Light Dependant Resistor*), dengan masing-masing properti model warna HSV. Dilakukan pada saat tertentu dan dengan Exposur Value yang tetap yaitu -2, dan pengaturan White Balance yang tetap pula yaitu kondisi *Daylight*. Terlihat di Gambar 4.4 pengaruh intesitas cahaya terhadap nilai Hue (HSV) tidak terlalu signifikan, dimana nilai Hue tetap berubah namun tidak bisa ditentukan apakah akan meningkat seiring bertambahnya intensitas cahaya atau sebaliknya. Seperti halnya pada subbab 4.1.1. berubahnya nilai Hue bergantung pada dimana nilai tersebut menjadi acuan.




**Gambar 4. 4** Grafik respon Hue terhadap Intensitas Cahaya

### 4.1.5. Intensitas Cahaya – Saturation

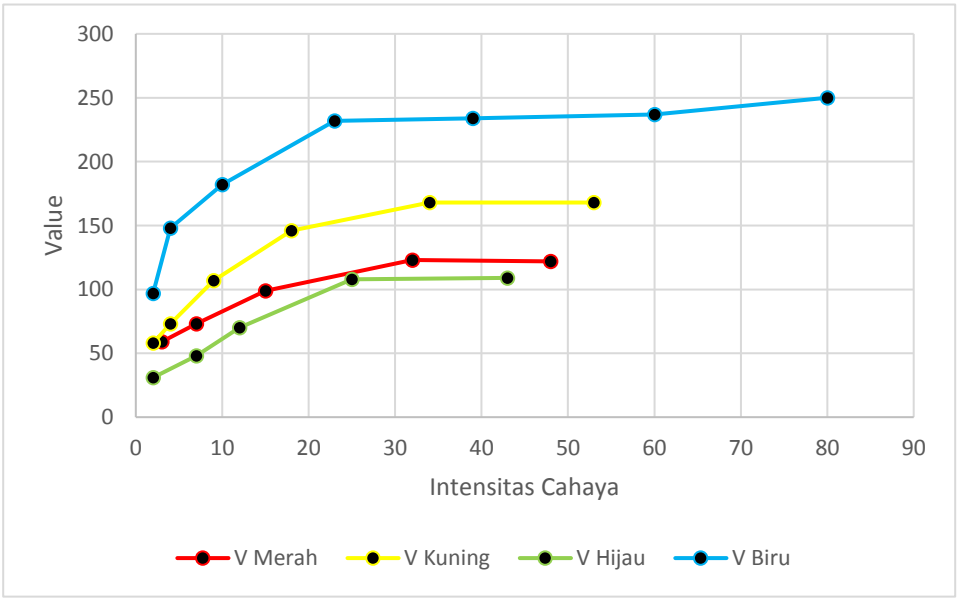
Pada Gambar 4.5. ditampilkan hasil pengambilan data yang menghubungkan nilai intensitas cahaya dengan nilai Saturation (HSV) pada saat waktu tertentu dan dengan nilai *exposure* yang tetap yaitu -2, dan dengan pengaturan *white balance* yang tetap, yaitu kondisi *daylight*. Terlihat di grafik untuk obyek berwarna merah dan kuning, semakin besar nilai intensitas cahaya maka semakin kecil nilai Saturation (HSV) yang terjadi. Namun, pada obyek percobaan berwarna hijau dan biru terdapat perbedaan, dimana nilai Saturation (HSV) yang terjadi cenderung tetap pada nilai atas (255) meskipun intensitas cahaya berubah. Hal ini dapat terjadi karena proses pengambilan data dilakukan pada *exposure* perangkat yang masih terlalu rendah dengan kondisi cahaya lingkungan yang tidak signifikan. Sehingga, perubahan nilai Saturation (HSV) pada obyek warna tersebut dimungkinkan tidak terjadi.




**Gambar 4. 5** Grafik respon Saturation terhadap Intensitas Cahaya

### 4.1.6. Intensitas Cahaya – Value

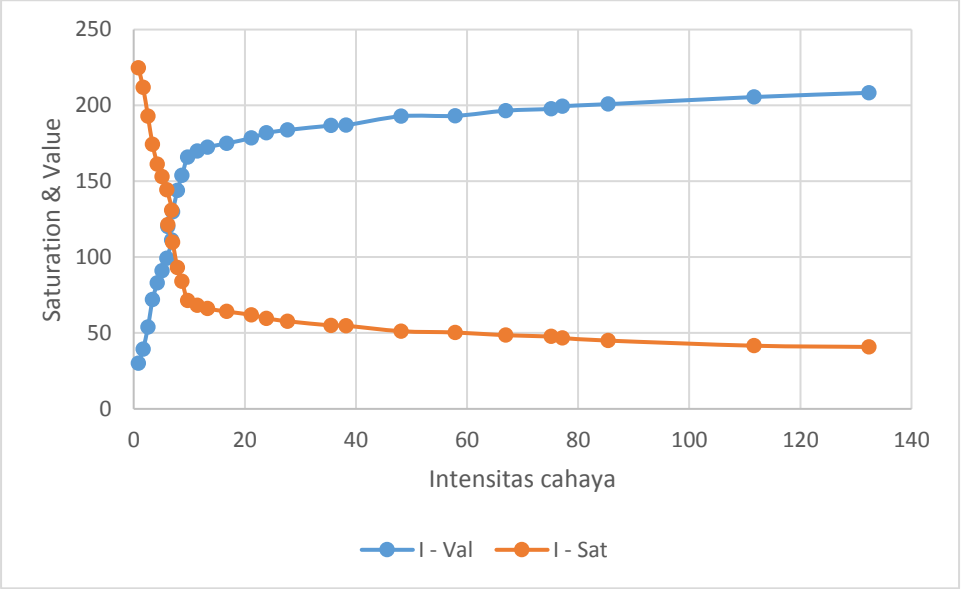
Pada Gambar 4.6 diperlihatkan hubungan antara respon Value (HSV) terhadap nilai intensitas cahaya dari sensor, dengan kondisi parameter nilai *exposure* dan *white balance* yang tetap. Dapat diamati dari grafik bahwa semakin terang atau tinggi intensitas cahaya, semakin besar pula nilai Value (HSV) dari suatu obyek. Sebagaimana ditunjukkan juga oleh kurva, semakin nilai Value (HSV) mendekati batas atasnya, maka cenderung semakin kecil tingkat perubahan atau *slope* yang terjadi.




**Gambar 4. 6** Grafik respon Value terhadap Intensitas Cahaya

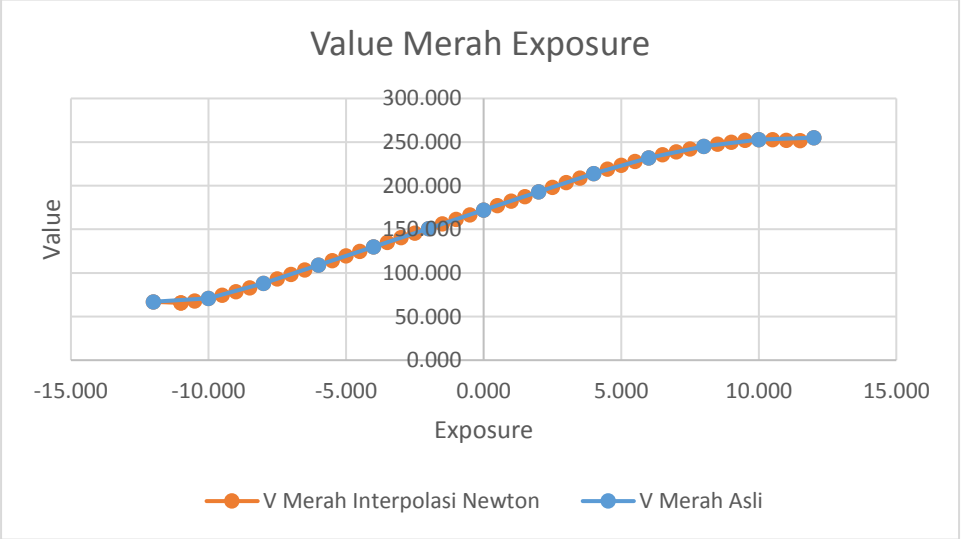
### 4.1.7. Intensitas Cahaya – Average Saturation & Value (Kamera)

Hasil data pada subbab sebelumnya menunjukkan bahwa intensitas cahaya memiliki pengaruh yang signifikan terhadap Saturation dan Value pada model warna HSV. Oleh karena itu, perlu diamati pula bagaimana pengaruh nilai rata-rata Saturation dan Value dari seluruh pixel yang ditangkap oleh kamera. Terlihat pada Gambar 4.7 bahwasanya nilai Saturation dan Value rata-rata dari seluruh pixel cenderung simetris satu sama lain.

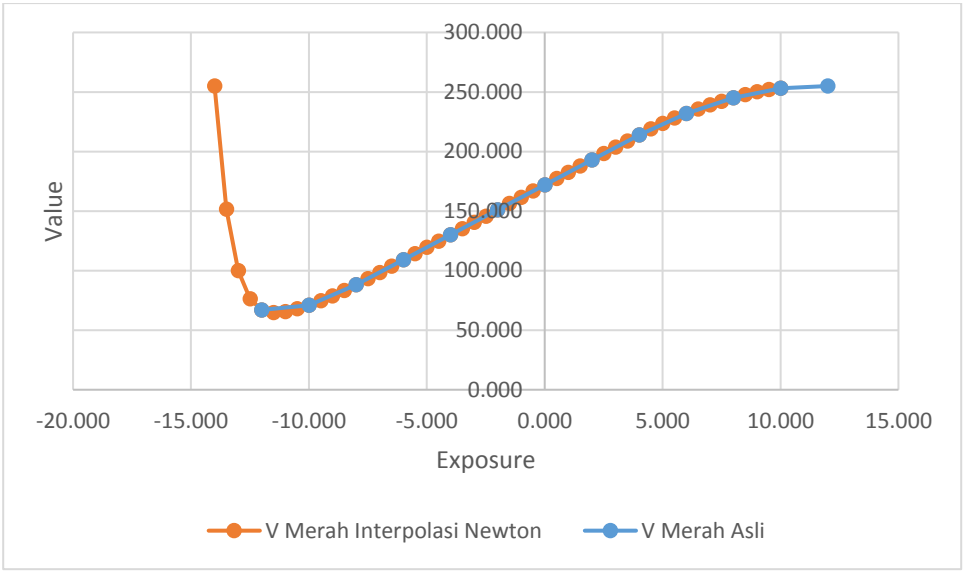



**Gambar 4. 7** Grafik respon Saturation& Value terhadap Intensitas Cahaya

### 4.1.8. Interpolasi Nilai *Exposure* – Value



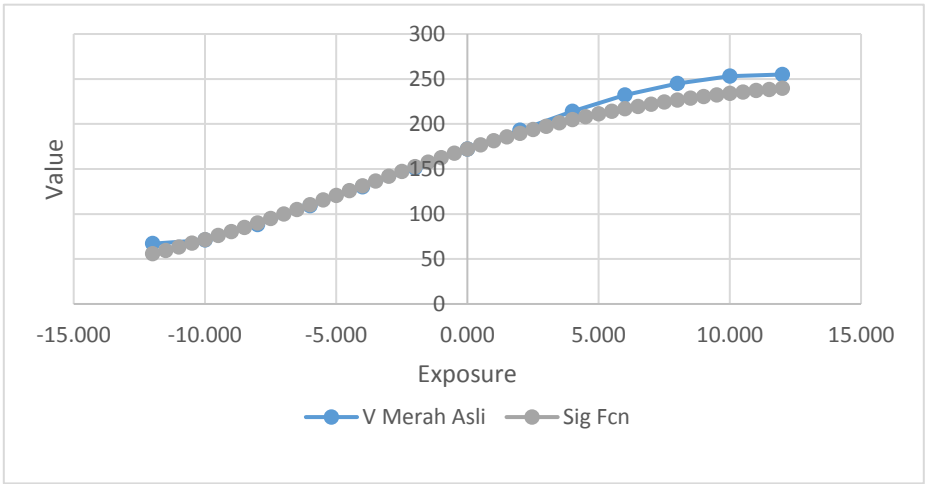

**Gambar 4. 8** Grafik hasil interpolasi value terhadap exposure




**Gambar 4. 9** Pergeseran dengan mengubah nilai x untuk obyek lain

Pada Gambar 4.8 ditunjukkan hasil pencocokan kurva pada Value (HSV) merah terhadap nilai *exposure* menggunakan interpolasi polynomial. Dari metode tersebut dihasilkan fungsi $y = 171.9 + 10.5*x + 0.065*x^2 + 0.0007735*x^3 + (-0.00382)*x^4 + (-0.0003141)*x^5 + 0.00002646*x^6 + 0.000002512*x^7 + -0.00000004662*x^8-0.000000008842*x^9$, dengan $y$ adalah Value (HSV) dan $x$ adalah nilai *exposure*. Hasil perhitungan menggunakan metode interpolasi dapat dilihat dalam lampiran. Cara ini kurang tepat untuk digunakan pada data EV-Value yang telah didapat karena jika kurva digeser untuk obyek warna lain, hasil akan cenderung bernilai salah, seperti terlihat dalam Gambar 4.9, Karena memang interpolasi tidak tepat untuk memperkirakan nilai diluar batas penentuannya.

### 4.1.9. Pencocokan Kurva Hyperbolic Tangent

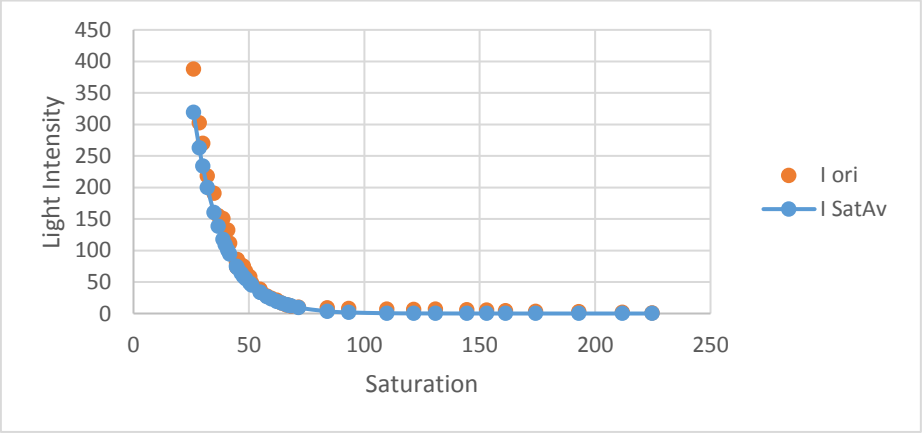



**Gambar 4. 10** Grafik fungsi hyperbolic tangent EV – Value

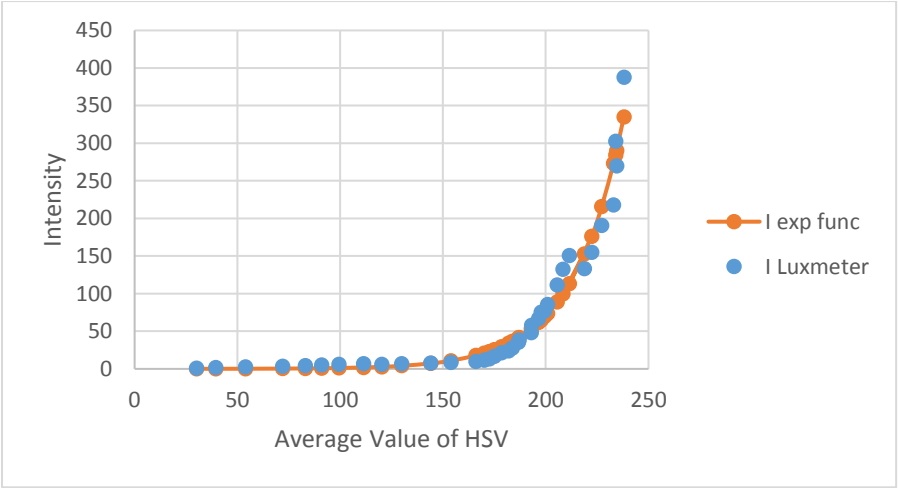
Metode pencocokan kurva lain yang coba dilakukan dalam skripsi ini ialah menggunakan Hyperbolic Tangent. Persamaan hyperbolic tangent disesuaikan gradien dan titik-titiknya dengan data yang telah didapat, seperti terlihat pada Gambar 4.9. Pada data hubungan nilai *exposure* dengan Value (HSV) obyek merah didapat fungsi *y(x)* =128*TANH(*x*0.084+0.368) + 127, dengan *y* adalah Value (HSV) dan *x* adalah nilai *exposure*. Melalui cara ini fungsi yang dihasilkan dapat digeser dengan mengurangkan atau menambahkan *x* dengan variabel tertentu, sehingga dapat disesuaikan dengan Value (HSV) obyek lain. Untuk hubungan nilai *exposure* dengan nilai Saturation juga dapat menggunakan cara ini. Fungsi yang didapat untuk nilai Saturation (HSV) ialah y = 127*-TANH(x*0.07+0.28) + 128, dengan *y* adalah Value (HSV) dan *x* adalah nilai *exposure*.

### 4.1.10. Exponential Curve Fitting

Data yang telah diperoleh pada Gambar 4.7 dapat digunakan untuk menentukan fungsi yang memperkirakan nilai intensitas cahaya melalui Saturation (HSV) atau Value (HSV) rata-rata seluruh pixel. Dari data yang didapat menunnjukkan kurva yang menyerupai fungsi eksponensial, yang memiliki persamaan dasar $y = e^x$. Oleh karena itu, metode pencocokan kurva yang sesuai ialah menggunakan fungsi yang sama. Dari proses pencocokan kurva menggunakan metode ini diperoleh fungsi $y = 0.0000055556*e^{(0.07802*(255-x)}$, dengan *y* adalah intensitas cahaya dan *x* adalah nilai Saturation (HSV), grafik dapat dilihat pada Gambar 4.11. Untuk perkiraan intensitas cahaya melalui rata-rata nilai Value (HSV), dapat menggunakan fungsi $y = 0.0004885*e^{(0.06007*x)}$, dengan *y* merupakan intensitas cahaya dan *x* merupakan nilai Value (HSV), grafik dapat dilihat dalam Gambar 4.12.




**Gambar 4. 11** Grafik fungsi eksponensial Saturation - I



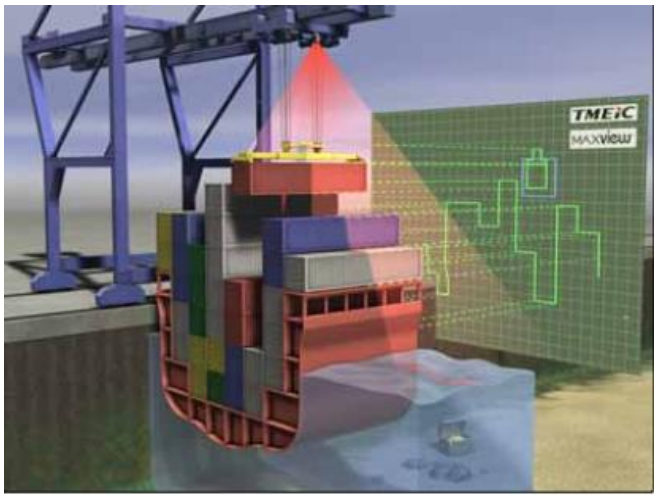

**Gambar 4. 12** Grafik fungsi eksponensial Value - I

## 4.2. Simulasi Penerapan

Pada sub-bab ini akan dibahas mengenai cara pendeteksian gerakan pada spreader pada sistem yang dicoba.

### 4.2.1. Peletakan Kamera dari Sistem

Pada sistem yang diusulkan, kamera diletakkan di sisi belakang troli, arah kamera menghadap ke penyebar, tempat yang sama seperti sensor laser TMEIC Maxview pada Gambar 4.13, namun agak sedikit ke belakang. Sudut pandang ini sama dengan tampilan operator pada crane yang dioperasikan secara manual. Melalui ini, rangkaian gambar atau video yang dihasilkan oleh kamera termasuk *spreader* dan kontainer dapat diamati sama dengan pandangan operator kabin, ditunjukkan oleh Gambar 4.14

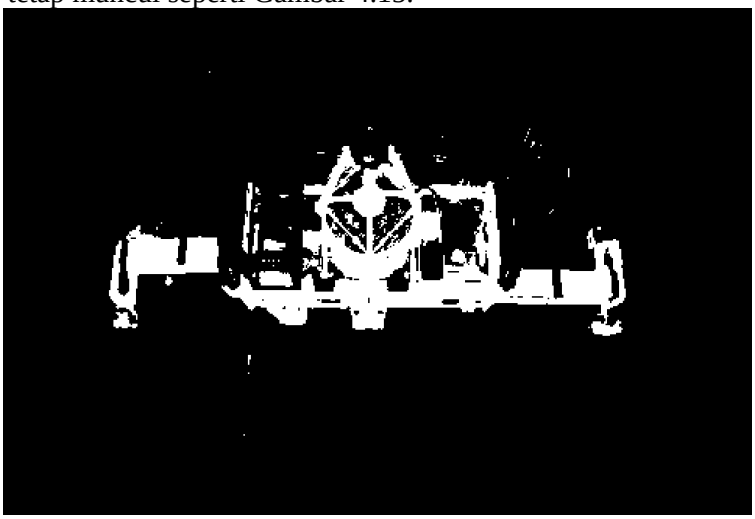



**Gambar 4. 13** Letak Laser pada crane TMEIC Maxview
(TMEIC Corporation, 2017)

**Gambar 4. 14** Sudut pandang operator kabin
(https://www.youtube.com/watch?v=vA0ejYk1s4k&t=101s)

### 4.2.2. Thresholding

Thresholding adalah cara termudah untuk mensegment objek yang diinginkan, dengan cara ini citra biner dapat dihasilkan [11]. Pada OpenCV tugas ini bisa dilakukan hanya dengan memanggil fungsi `range`. Citra ambang batas HSV pada spreader Gambar 4.14 dapat ditunjukkan sebagai Gambar 4.15. Masalah dapat muncul saat rentang nilai batas terlalu sempit objek cenderung tidak terdeteksi, namun bila rentangnya terlalu lebar, piksel yang tidak diinginkan mungkin tetap muncul seperti Gambar 4.15.



**Gambar 4. 15** Thresholed Image

### 4.2.3. Median Filter (Smoothing)

Ada beberapa cara untuk mengurangi piksel yang tidak diinginkan dari gambar. Cara sederhana untuk melakukan tugas ini adalah dengan metode smoothing. Jenis operator yang paling umum digunakan adalah filter linier, dimana nilai piksel keluaran ditentukan sebagai jumlah tertimbang dari nilai piksel. Berikut formula dari convolution filter:

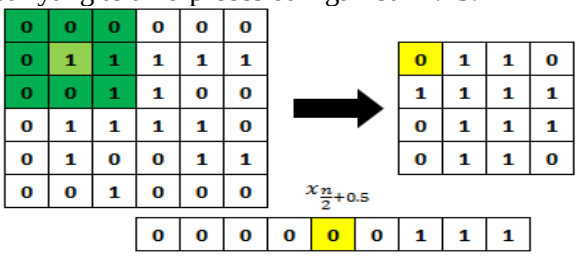
$$g = f \star h \tag{17}$$

$$g(i,j) = \sum_{k,l} f(i-k, j-l)h(k,l) \tag{18}$$

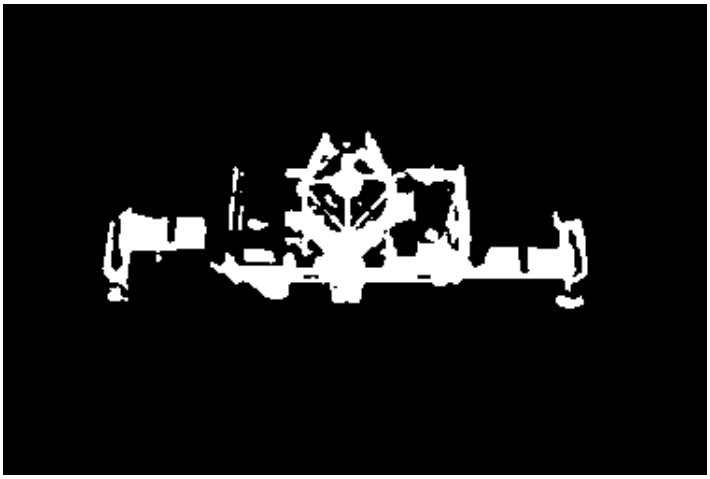
Ada berbagai macam filter, seperti Normalized Box Filter, Gaussian Filter, Median Filter, Bilateral Filter. Skripsi ini menggunakan Median Filter, karena cocok untuk noise seperti pada Gambar 4.15 yang merupakan citra biner. Cara ini tidak mensamarkan tepi seperti Gaussian Filter. Filter h (k, l) dari Filter Median harus mengikuti persamaan ini:

$$Me = \uparrow x_{\frac{n}{2}+0.5} \tag{19}$$

Operasi menggunakan persamaan tersebut karena matriks kernel selalu berupa jumlah ganjil. Seperti ditunjukkan Gambar 4.16, masukan matriks biner 6x6 dilingkupi oleh Median Filter bermatrik 3x3, menghasilkan nilai pada koordinat piksel anchor (ditunjukkan dengan highlight kuning). Di OpenCV Median Blur dilakukan dengan memanggil fungsi `medianBlur`. Gambar 4.17 merupakan gambar yang telah diproses dari gambar 4.15.



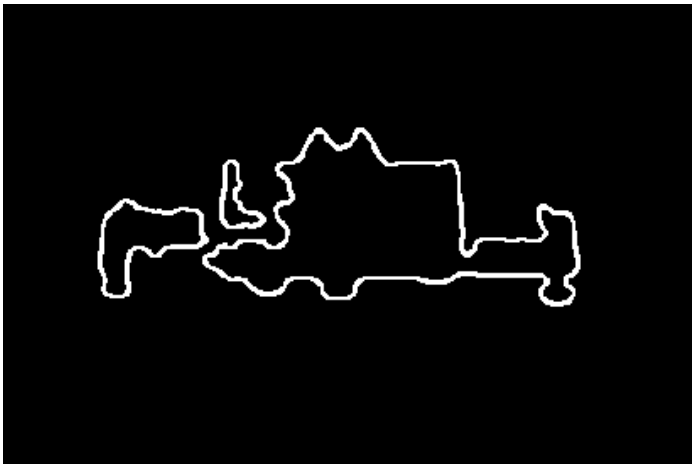

**Gambar 4. 16** Ilustrasi median filtering



**Gambar 4. 17** Thresholded Image dengan Median Filter

### 4.2.4 Find Contours

Kontur merupakan kurva yang menggabungkan semua titik secara kontinu. Menemukan kontur adalah hal yang berguna dalam analisis bentuk dan pengenalan objek [11]. Skripsi ini menggunakan fungsi `findContours` OpenCV untuk mendapatkan kontur dari gambar biner. Seperti ditunjukkan pada gambar 4.18, cara kerja fungsi ini ialah dengan menemukan pixel putih dari backgound hitam. Gambar biner dari Gambar 4.16 diubah menjadi tiga grup garis isolasi, disebut kontur. Setiap kontur memiliki fitur indeks dan hierarki yang membuat gambar lebih mudah dianalisis.



**Gambar 4. 18** Keluaran kontur dari image biner

### 4.2.5. Contour Area

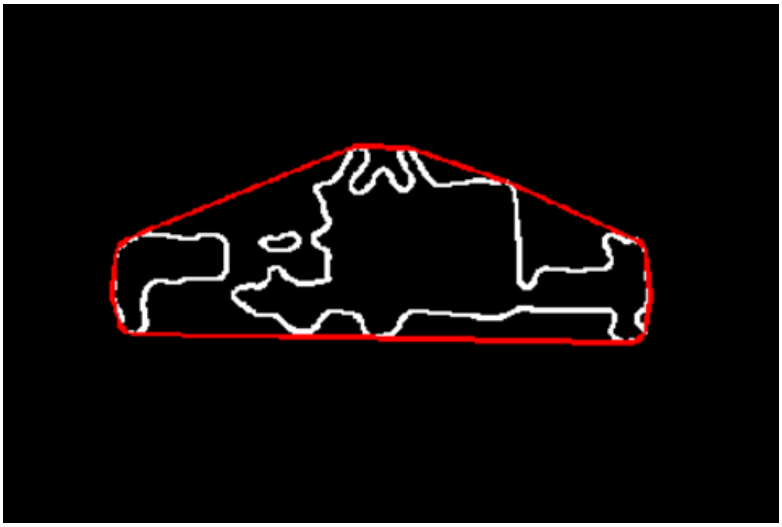
Memperkirakan area kontur diperlukan untuk proses lebih lanjut, seperti perkiraan posisi relatif *spreader* terhadap troli, apakah akan turun atau naik. Untuk mendapatkan nilai area kontur, dalam tugas akhir ini menggunakan fungsi `contourArea` dari library OpenCV. Fungsi ini menghitung daerah dengan memanfaatkan teorema Green [11]. Hal ini memberikan hubungan antara integral garis di sekitar kurva tertutup sederhana (kontur) C dan integral ganda di atas bidang bidang D yang dibatasi oleh kontur tersebut, jika L dan M adalah fungsi (x, y) yang didefinisikan Pada daerah terbuka yang mengandung D dan memiliki turunan parsial kontinyu di sana, maka relasinya ditunjukkan oleh persamaan berikut.

$$\oint_C Ldx + Mdy = \iint_D \left(\frac{\partial M}{\partial x} - \frac{\partial L}{\partial y}\right) dxdy \tag{20}$$

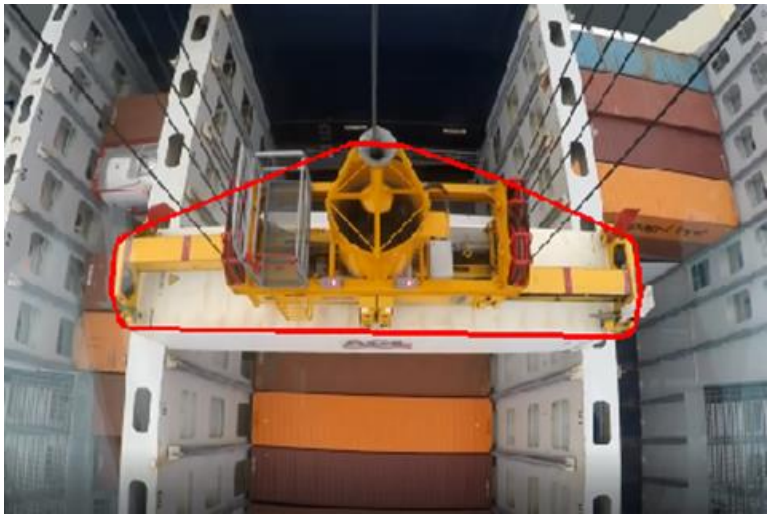
### 4.2.6. Convex Hull Area

Untuk menyatukan dan menyelubungi semua kontur yang telah dihasilkan, maka fungsi `convexHull` dari OpenCV ddapat digunakan. Convex hull adalah bentuk penyatuan kontur yang dihubungkan garis dengan menyambungkan titik

kontur terluar. Dalam hal ini convex hull dapat divisualisasikan sebagai bentuk yang menghubungkan kontur yang telah didapat sebelumnya, dapat dilihat pada Gambar 4.19.



(a)



(b)

**Gambar 4. 19** Convex hull dari kontur ditampilkan dalam image biner (a) dan input (b)

### 4.2.7. Find Spreader Edge

Convex hull pada Gambar 4.10 ikut menghitung kontur drum kabel, bentuk ke atas meruncing. Untuk menghindari hal ini dan agar deteksi menjadi lebih mudah, setiap titik tepi penyebar harus diperoleh. Makalah ini menggunakan beberapa algoritma untuk mendapatkan setiap titik tepi yang dapat berguna untuk proses selanjutnya, yaitu upper-uppermost (LU), right-uppermost (RU), left-bottommost (LB), dan right-bottommost (RB). Setiap titik kontur P dibandingkan

dengan sisi berlawanan pada ujung bingkai, hitung panjang antara dua titik itu, dan titik terpanjang dari ujung bingkai dapat dianggap sebagai tepi yang diinginkan. Sebagai contoh, dengan panjang frame 480px, lebar 320px, dan L sebagai panjang titik yang diperiksa P ke ujung frame yang berlawanan E, cara ini bisa diformulasikan sebagai berikut:

$$L(P,E) = \sqrt[2]{(Px - Ex)^2 + (Py - Ey)^2} \tag{21}$$

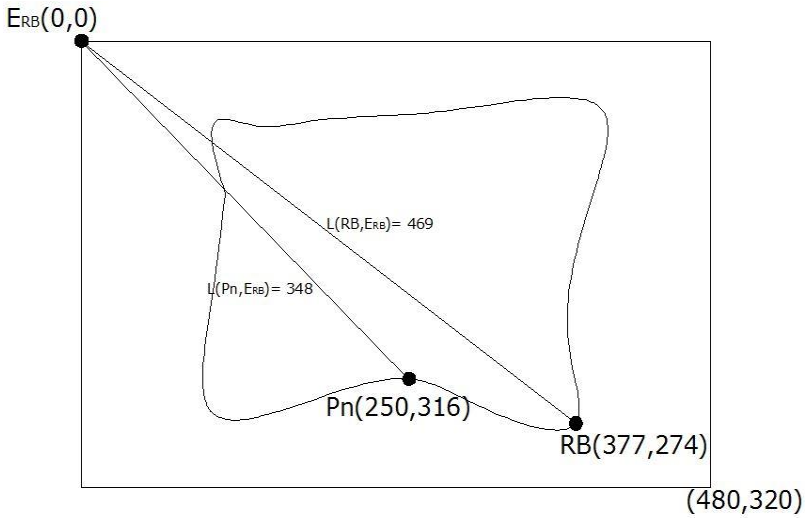
$$RB_{(x,y)} = \begin{cases} P, & if\ L(P,E_{RB}) > L(RB,E_{RB}) \\ RB, & otherwise \end{cases} \tag{22}$$

$$LB_{(x,y)} = \begin{cases} P, & if\ L(P,E_{LB}) > L(RB,E_{LB}) \\ LB, & otherwise \end{cases} \tag{23}$$

$$RU_{(x,y)} = \begin{cases} P, & if\ L(P,E_{RU}) > L(RB,E_{RU}) \\ RU, & otherwise \end{cases} \tag{24}$$

$$LU_{(x,y)} = \begin{cases} P, & if\ L(P,E_{LU}) > L(RB,E_{LU}) \\ LU, & otherwise \end{cases} \tag{25}$$

Pada persamaan tersebut, sisi berlawanan dari RB kanan-paling bawah adalah ERB (0,0), sedangkan sisi berlawanan LU, RU, LB adalah ELU (480.320), ERU (0,320), ELB 480,0). Dengan cara ini diilustrasikan oleh Gambar 10, untuk menemukan RB kanan-bawah, panjang masing-masing titik ke ERB frame frame yang berlawanan (menggunakan Persamaan) dibandingkan, nilai terpanjang yang dianggap sebagai BPR, ilustrasi ditunjukkan oleh Gambar 4. 20. Implementasi algoritma ini ditunjukkan pada Gambar 4. 21.

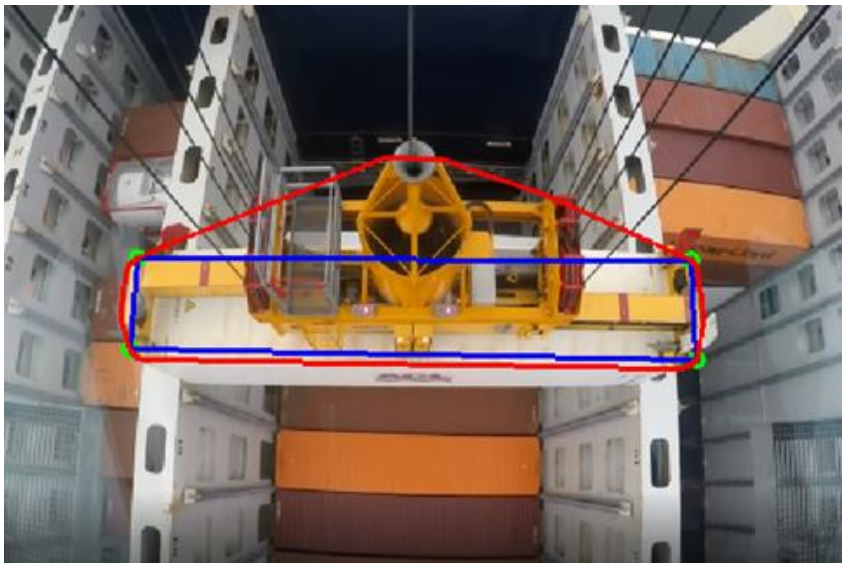



**Gambar 4. 20** Ilustrasi algoritma pencari sisi

**Gambar 4. 21** Penerapan algoritma pencari sisi

### 4.2.8. Making Line

Setelah menemukan titik-titik pada ujung spreader, bisa dibuat garis dari satu sisi ke tepi lainnya. Untuk menarik garis, tugas kahir ini menggunakan fungsi OpenCV `line`, dilakukan dengan menggambar garis dari satu titik yang ditentukan ke titik lain. Karena titik tepi sudah diperoleh, garis bisa dibuat, ditunjukkan seperti Gambar 4. 22.



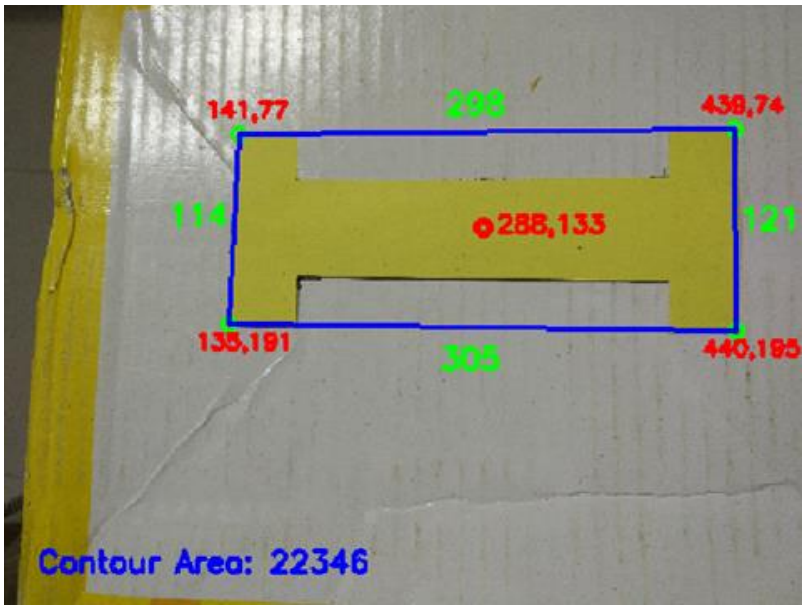
(a)


(b)

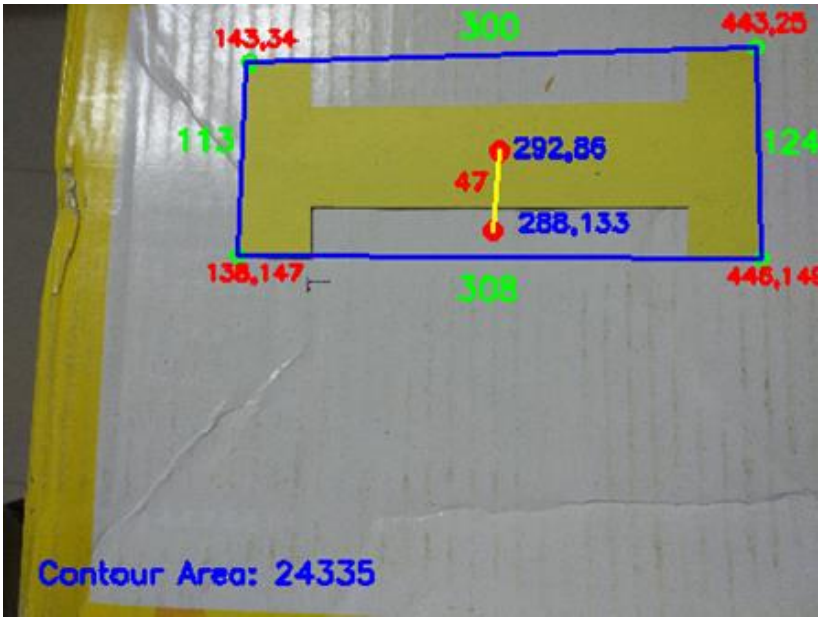
**Gambar 4. 22** Bounded line dari satu titik ke titik lain

### 4.2.9. Center Point Displacement

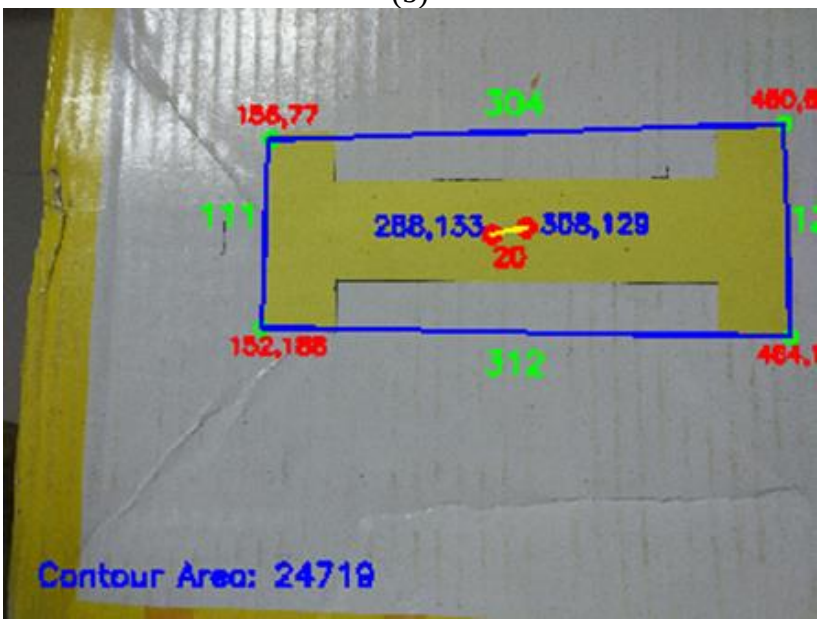
Dengan dilakukannya perhitungan jarak titik tengah objek ke titik referensi, dapat dideteksi perpindahan objek. Seperti ditunjukkan pada Gambar 4. 23 (b) dan (c), dari pandangan kamera obyek telah berpindah sejauh 47 pixel pada (b) dan 20 pixel pada (c) dari titik referens (a). Dengan adanya garis yang menghubungkan titik tersebut, dapat dideteksi pula arah perpindahan suatu obyek. Perpindahan ini nantinya dapat bermanfaat untuk menganalisis tingkat *sway* pada *spreader*.




(a)




(b)



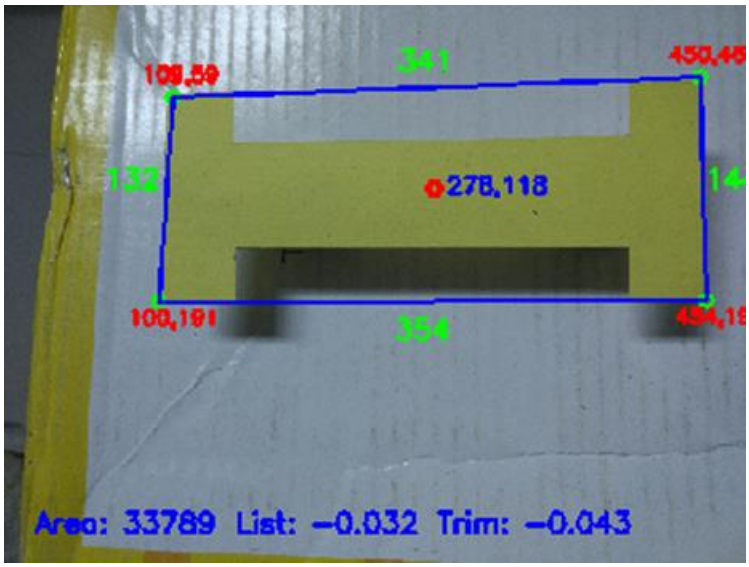

(c)

**Gambar 4. 23** Simulasi perpindahan posisi dari referensi (a) ke posisi (b) dan (c)

### 4.2.10. Vertical Displacement

Area kontur objek bisa mewakili jarak ke kamera. Jika area menjadi semakin besar, dapat diperkirakan bahwa objek semakin dekat dengan kamera, dan jika area menjadi semakin kecil, dapat disimpulkan bahwa objek semakin jauh

dengan kamera, ilustrasi dapat dilihat dari perbandingan Gambar 4. 23 (a) dengan Gambar 4. 24. Pada kondisi tertentu, sebenarnya titik pusat objek juga bisa mewakili jarak vertikal jika kamera tidak tepat mengarah ke tengah, tapi cara ini bisa salah persepsi jika objeknya cenderung bergerak secara horisontal. Karena kamera berada di troli yang menghadap ke *spreader*, jarak antara kamera dan troli bisa diestimasi.

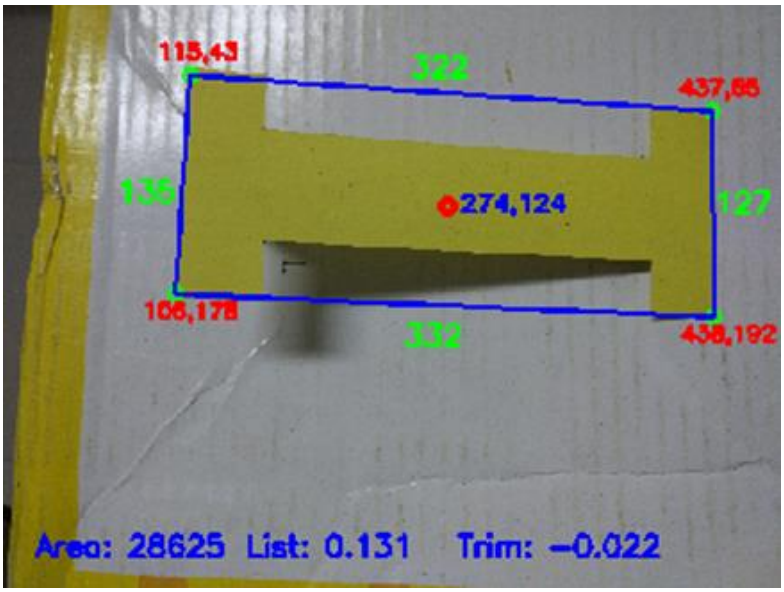



**Gambar 4. 24** Simulasi gerakan vertikal

### 4.2.11. List

List dapat diamati dengan membandingkan panjang sisi yang bersangkutan, untuk mendapatkan rasionya. Perbandingan ini kemudian dihitung untuk mendapatkan perbedaan dengan rasio referensi. Algoritma ini diimplementasikan pada Gambar 4.25 yang mana dibandingkan dengan referensi pada Gmbar 4. 23 (a), dan melalui percobaan hal ini dapat mewakili perubahan list. Algoritma ini menyatakan bahwa nilai list yang menjauh dari nol adalah derajat list yang lebih besar. Untuk mendapatkan nilai standar yang dapat diimplementasikan pada aplikasi spreader crane, metode ini harus dianalisis lebih lanjut baik dengan metode eksperimen nyata atau teori fisika.

$$List = \left(\frac{L1_0}{L2_0} - \frac{L1}{L2}\right) * \text{scalar} \quad (26)$$

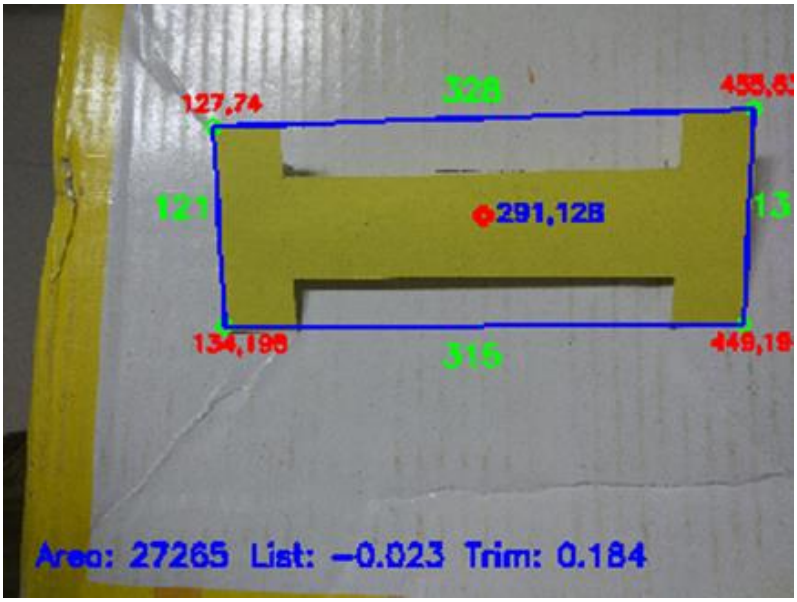



**Gambar 4. 25** Simulasi list

### 4.2.12. Trim

Sama seperti list, trim diamati oleh kedua panjang sisi yang bersangkutan untuk mendapatkan rasio. Algoritma diimplementasikan pada Gambar 4. 26 dengan perbandingannya dari Gambar 4. 23 (a). Nilai trim yang menjauh dari nol menyatakan bahwa tingkat trim yang lebih besar terjadi.

$$Trim = \left(\frac{L3_0}{L4_0} - \frac{L3}{L4}\right) * \text{scalar} \quad (27)$$




**Gambar 4. 26** Simulasi trim

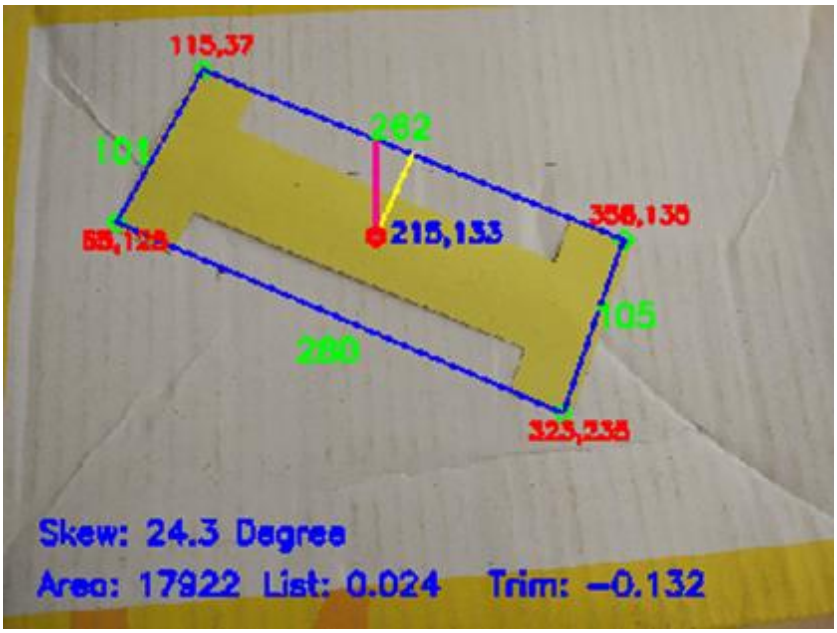
### 4.2.13. Skewness

Untuk mengamati kemiringan objek atau skewness, sudut harus diperoleh terlebih dahulu. Kemudian, perbedaan pada sudut referensi dapat dihitung, seperti yang dirumuskan dalam persamaan berikut.

$$\theta = \sin^{-1}\frac{y}{r}\,\text{rad} = \sin^{-1}\frac{y}{r} * \frac{360}{2\pi}\,\text{degree} \quad (28)$$

$$\text{Skewness} = \theta_1 - \theta_0 \quad (29)$$

Implementasi algoritma ini ditunjukkan pada percobaan Gambar 4.27 dengan perbandingannya dri Gambar 4.23 (a). Ini mungkin berguna untuk diterapkan pada *spreader* crane, karena hanya menggunakan satu sensor, berbeda dengan sensor inframerah yang paling tidak membutuhkan dua set sensor untuk mendeteksi ini.




**Gambar 4. 27** Simulasi skewness

*Halaman ini sengaja dikosongkan*

# BAB V
# KESIMPULAN DAN SARAN

## 5.1. Kesimpulan

Berdasarkan hasil dan pembahasan dari tugas akhir yang telah dibuat, dapat diambil beberapa kesimpulan, yaitu:

1. Peningkatan intensitas cahaya mengakibatkan meningkatnya nilai Saturation dan menurunnya nilai Value pada model warna HSV. Sedangkan pengaruh cahaya terhadap nilai Hue bergantung pada letak dimana Hue pertama kali diacu.
2. Hubungan antara tingkat *Exposure* obyek dengan nilai Value (HSV) dapat diperkirakan dengan fungsi $y = 128*TANH(x*0.084+0.368) + 128$, dimana $y$ merupakan Value (HSV) dan $x$ merupakan tingkat *exposure*. Hubungan antara tingkat *Exposure* obyek dengan nilai Saturation (HSV) dapat diperkirakan dengan fungsi $y = 127*-TANH(x*0.07+0.28) + 128$, yang mana $y$ merupakan Value (HSV) dan $x$ merupakan tingkat *exposure*.
3. Hubungan antara Intensitas Cahaya dengan rata-rata Value (HSV) seluruh pixel dapat diperkirakan dengan fungsi eksponensial $y = 0.0004885*e^{(0.06007*X)}$**.** Dimana $y$ = Intensitas Cahaya dan $x$ = rata-rata nilai saturation. Sedangkan, antara Intensitas cahaya dengan Saturation (HSV) dapat diperkirakan dengan fungsi $y = 0.0000055556*e^{(0.07802*(255-X)}$ dimana $y$ = intensitas cahaya dan $x$ = nlai Saturation HSV. Melalui persamaan ini dapat diketahui respon perubahan cahaya semata-mata hanya dengan mengetahui rata-rata nilai Saturation ataupun Value dari keseluruhan pixel.
4. Penerapan *computer vision* untuk *crane spreader* dapat dilakukan dengan menganalisa garis dan titik yang melingkupi crane untuk deteksi tingkat *trim* dan *list*. Area kontur untuk perpindahan posisi vertikal. Titik pusat untuk perpindahan horizontal, juga untuk parameter swaying. Dan perpindahan sudut *spreader* untuk tingkat skew.

## 5.2. Saran

Saran untuk penelitian selanjutnya diharapkan data gerakan yang dihasilkan menggunakan pengolahan citra pada skripsi ini, dapat dianalisa atau dilakukan eksperimen sebagai parameter input kontrol untuk penerapan crane.

*Halaman ini sengaja dikosongkan*

# DAFTAR PUSTAKA


[1] R. C. Gonzalez and R. E. Woods, Digital Image Processing, Prentince Hall, 2008.

[2] W. Burger and M. J. Burge, Digital Image Processing: An Algorithmic Approach Using Java, Springer-Verlag, 2008.

[3] M. Sonka, V. Hlavac and R. Boyle, Image Processing, Analysis, and Machine Vision, Ontario (CA): Thomson, 2008.

[4] D. A. Forsyth and J. Ponce, Computer Vision: A Modern Approach, New Jersey, NJ (USA): Prentince Hall, 2003.

[5] G. Bradski and A. Kaehler, Learning OpenCV: Computer Vision with the OpenCV Library, 1st ed., Sebastopol, CA: O'Reilly Media, 2008.

[6] mhstekkomp, "mhstekkomp," [Online]. Available: https://mhstekkomp.wordpress.com/2011/05/07/representasi-model-warna-rgb-menggunakan-hsl-dan-hsv/. [Accessed 6 February 2017].

[7] D. Travis, Color Displays, Theory and Practice, Cambridge, Massachusetss: Academic Press, 1991.

[8] Wikipedia, "Curve Fitting," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Curve_fitting#cite_note-1. [Accessed 20 7 2017].

[9] Wikipedia, "Interpolation," Wikipedia, 19 May 2017. [Online]. Available: https://en.wikipedia.org/wiki/Interpolation. [Accessed 28 July 2017].

[10] Mathworks, "Curve Fitting Toolbox," The Mathworks, inc, [Online]. Available: https://www.mathworks.com/products/curvefitting.html. [Accessed 28 July 2017].

[11] A. Huaman and T. Tsemelis, "OpenCV Imgproc Module," OpenCV, [Online]. Available: http://docs.opencv.org/3.1.0/d7/da8/tutorial_table_of_content_imgproc.html. [Accessed 15 July 2017].

[12] R. Szeliski, Computer Vision Algorithms and Applications, New York: Springer-Verlag, 2010.

[13] Wikipedia, "Crane (machine)," Wikipedia, 20 July 2017. [Online]. Available: https://en.wikipedia.org/wiki/Crane_(machine). [Accessed 28 July 2017].

[14] TMEIC, "The Maxview Smart Landing System," TMEIC Corporation, Tokyo, 2017.

[15] K. Obata, K. Uchida, T. Chikura, H. Yhosikawa and T. Monzen, "Automated Transfer Crane," *Mitsubishi Heavy Industries, Ltd. Technical Review,* vol. 40, no. 2, pp. 1-5, January 2003.

[16] J. Hollingsworth, "A Beginner's Guide for Manual Controls in iPhone Photography: Exposure," Camera+, 24 October 2014. [Online]. Available: http://snapsnapsnap.photos/a-beginners-guide-for-manual-controls-in-iphone-photography-exposure/. [Accessed 28 July 2017].

*Halaman ini sengaja dikosongkan*

# LAMPIRAN

- **Data HSV-EV**

| No. | Exp | WB | Objek 1 (Merah) | | | | | | Objek 2 (Kuning) | | | | | | Objek 4 (Hijau) | | | | | | Objek 6 (Biru) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Hmid1 | Smid1 | Vmid1 | Hmid2 | Smid2 | Vmid2 | Hmid1 | Smid1 | Vmid1 | Hmid2 | Smid2 | Vmid2 | Hmid1 | Smid1 | Vmid1 | Hmid2 | Smid2 | Vmid2 | Hmid1 | Smid1 | Vmid1 | Hmid2 | Smid2 | Vmid2 |
| 1 | -12 | DL | 4 | 236 | 67 | 4.5 | 253.5 | 53 | 59 | 255 | 51 | 59.5 | 253.5 | 58.5 | 92 | 255 | 43 | 91 | 255 | 33 | 160 | 255 | 36 | 161.5 | 255 | 22.5 |
| 2 | -10 | DL | 4 | 231 | 71 | 4.5 | 251.5 | 69 | 58 | 254 | 72 | 58.5 | 253 | 70.5 | 95 | 255 | 55 | 91 | 255 | 42 | 156 | 255 | 42 | 159.5 | 255 | 26.5 |
| 5 | -8 | DL | 5 | 225 | 88 | 4.5 | 247.5 | 84.5 | 56 | 251 | 93 | 56.5 | 252 | 83 | 97 | 255 | 64 | 91.5 | 255 | 52.5 | 152 | 254 | 50 | 157.5 | 254 | 33 |
| 7 | -6 | DL | 7 | 218 | 109 | 5.5 | 242.5 | 100 | 55 | 246 | 114 | 55.5 | 249 | 99 | 99 | 255 | 76 | 93.5 | 255 | 64 | 150 | 250 | 60 | 155.5 | 251.5 | 43 |
| 9 | -4 | DL | 7 | 211 | 130 | 7.5 | 237 | 117.5 | 54 | 239 | 135 | 55 | 244.5 | 118 | 101 | 255 | 88 | 95.5 | 253 | 75 | 148 | 245 | 73 | 152 | 247 | 55.5 |
| 11 | -2 | DL | 7 | 203 | 151 | 7.5 | 228.5 | 136.5 | 54 | 232 | 156 | 54.5 | 240 | 137 | 103 | 255 | 110 | 97.5 | 250 | 91 | 148 | 238 | 89 | 149.5 | 241.5 | 72.5 |
| 13 | 0 | DL | 7 | 194 | 172 | 8 | 219.5 | 156 | 53 | 224 | 177 | 53.5 | 235.5 | 157.5 | 104 | 255 | 129 | 101.5 | 245 | 109 | 147 | 230 | 110 | 148.5 | 236 | 90 |
| 15 | 2 | DL | 7 | 184 | 193 | 8 | 209.5 | 175.5 | 52 | 216 | 198 | 52.5 | 230.5 | 177.5 | 106 | 253 | 149 | 103.5 | 240 | 129 | 146 | 221 | 131 | 148 | 229.5 | 108 |
| 17 | 4 | DL | 8 | 173 | 214 | 8.5 | 198 | 193 | 51 | 207 | 219 | 52 | 226 | 197.5 | 106 | 249 | 170 | 105 | 235 | 147.5 | 146 | 212 | 152 | 147.5 | 221 | 127.5 |
| 19 | 6 | DL | 8 | 161 | 232 | 9.5 | 184.5 | 209.5 | 51 | 197 | 239 | 51 | 220.5 | 216.5 | 108 | 243 | 189 | 105.5 | 230 | 168.5 | 145 | 203 | 173 | 147.5 | 209.5 | 149 |
| 21 | 8 | DL | 9 | 148 | 245 | 10 | 170 | 222 | 50 | 186 | 253 | 49.5 | 213.5 | 231.5 | 108 | 236 | 209 | 105.5 | 224 | 196 | 145 | 194 | 194 | 146.5 | 198 | 169.5 |
| 23 | 10 | DL | 10 | 132 | 253 | 13 | 152.5 | 229 | 46 | 172 | 255 | 47 | 200 | 240 | 110 | 228 | 229 | 106 | 216 | 215 | 144 | 182 | 214 | 143.5 | 184 | 190 |
| 25 | 12 | DL | 15 | 111 | 255 | 13.5 | 131 | 231.5 | 43 | 156 | 255 | 46.5 | 178.5 | 246.5 | 111 | 217 | 244 | 109 | 200 | 230.5 | 141 | 169 | 229 | 143 | 168.5 | 206 |

| No. | Exp | WB | Objek 1 (Merah) | | | | | | Objek 2 (Kuning) | | | | | | Objek 4 (Hijau) | | | | | | Objek 6 (Biru) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Hmin | Hmax | Smin | Smax | Vmin | Vmax | Hmin | Hmax | Smin | Smax | Vmin | Vmax | Hmin | Hmax | Smin | Smax | Vmin | Vmax | Hmin | Hmax | Smin | Smax | Vmin | Vmax |
| 1 | -12 | DL | 0 | 9 | 252 | 255 | 27 | 79 | 55 | 64 | 252 | 255 | 42 | 75 | 85 | 97 | 255 | 255 | 16 | 50 | 153 | 170 | 255 | 255 | 5 | 40 |
| 2 | -10 | DL | 0 | 9 | 248 | 255 | 38 | 100 | 55 | 62 | 251 | 255 | 46 | 95 | 82 | 100 | 255 | 255 | 22 | 62 | 149 | 170 | 255 | 255 | 7 | 46 |
| 5 | -8 | DL | 0 | 9 | 240 | 255 | 48 | 121 | 54 | 59 | 249 | 255 | 51 | 115 | 82 | 101 | 255 | 255 | 31 | 74 | 147 | 168 | 253 | 255 | 11 | 55 |
| 7 | -6 | DL | 0 | 11 | 230 | 255 | 60 | 140 | 51 | 60 | 243 | 255 | 64 | 134 | 86 | 101 | 255 | 255 | 42 | 86 | 146 | 165 | 248 | 255 | 19 | 67 |
| 9 | -4 | DL | 2 | 13 | 219 | 255 | 76 | 159 | 51 | 59 | 234 | 255 | 84 | 152 | 88 | 103 | 251 | 255 | 51 | 99 | 146 | 158 | 239 | 255 | 29 | 82 |
| 11 | -2 | DL | 2 | 13 | 208 | 249 | 95 | 178 | 52 | 57 | 225 | 255 | 104 | 170 | 89 | 106 | 245 | 255 | 61 | 121 | 144 | 155 | 228 | 255 | 43 | 102 |
| 13 | 0 | DL | 2 | 14 | 197 | 242 | 115 | 197 | 51 | 56 | 216 | 255 | 125 | 190 | 95 | 108 | 235 | 255 | 75 | 143 | 144 | 153 | 217 | 255 | 57 | 123 |
| 15 | 2 | DL | 2 | 14 | 184 | 235 | 135 | 216 | 51 | 54 | 206 | 255 | 146 | 209 | 97 | 110 | 225 | 255 | 93 | 165 | 144 | 152 | 206 | 253 | 72 | 144 |
| 17 | 4 | DL | 2 | 15 | 170 | 226 | 153 | 233 | 49 | 55 | 197 | 255 | 167 | 228 | 102 | 108 | 215 | 255 | 110 | 185 | 144 | 151 | 195 | 247 | 88 | 167 |
| 19 | 6 | DL | 3 | 16 | 154 | 215 | 172 | 247 | 49 | 53 | 186 | 255 | 188 | 245 | 102 | 109 | 205 | 255 | 131 | 206 | 144 | 151 | 183 | 236 | 108 | 190 |
| 21 | 8 | DL | 3 | 17 | 136 | 204 | 191 | 253 | 48 | 51 | 172 | 255 | 208 | 255 | 102 | 109 | 194 | 254 | 165 | 227 | 143 | 150 | 171 | 225 | 130 | 209 |
| 23 | 10 | DL | 3 | 23 | 116 | 189 | 203 | 255 | 44 | 50 | 154 | 246 | 225 | 255 | 100 | 112 | 183 | 249 | 187 | 243 | 136 | 151 | 158 | 210 | 152 | 228 |
| 25 | 12 | DL | 6 | 21 | 96 | 166 | 208 | 255 | 42 | 51 | 131 | 226 | 238 | 255 | 104 | 114 | 172 | 228 | 206 | 255 | 135 | 151 | 144 | 193 | 168 | 244 |

- **Data Hubungan HSV rata-rata pixel dengan Intensitas Cahya**

| HueVal | SatVal | AvVal | I1 | I2 | ExpVal |
|---|---|---|---|---|---|
| 28 | 26 | 238 | 387.8 | 663 | -2 |
| 30 | 28.5 | 234 | 302.41 | 518 | -2 |
| 31.46 | 30 | 234.5 | 269.96 | 462 | -2 |
| 32.9 | 32 | 233 | 217.88 | 374 | -2 |
| 34.38 | 34.82 | 227.23 | 190.73 | 326 | -2 |
| 35.55 | 36.7 | 222.27 | 155.04 | 268 | -2 |
| 38 | 38.78 | 211.4 | 150.62 | 252 | -2 |
| 37.15 | 39.77 | 218.77 | 133 | 235 | -2 |
| 38.77 | 40.83 | 208.3 | 132.33 | 227 | -2 |
| 39.42 | 41.64 | 205.55 | 111.66 | 192 | -2 |
| 39.65 | 45 | 200.83 | 85.376 | 145 | -2 |
| 40 | 46.7 | 199.5 | 77.18 | 130 | -2 |
| 40.26 | 47.7 | 197.77 | 75.136 | 124 | -2 |
| 40.4 | 48.54 | 196.55 | 66.944 | 109 | -2 |
| 41.3 | 50.35 | 193.14 | 57.888 | 93 | -2 |
| 41.7 | 51.2 | 192.9 | 48.16 | 78 | -2 |
| 41.9 | 54.8 | 187 | 38.255 | 61 | -2 |
| 41.9 | 55 | 186.7 | 35.52 | 56 | -2 |
| 41.6 | 57.77 | 183.8 | 27.664 | 45 | -2 |
| 42.18 | 59.66 | 181.86 | 23.9 | 37 | -2 |
| 42.55 | 62 | 178.5 | 21.168 | 32 | -2 |
| 42.4 | 64.14 | 175 | 16.736 | 25 | -2 |
| 41.96 | 66.16 | 172.5 | 13.311 | 21 | -2 |
| 41.7 | 68.3 | 170 | 11.44 | 17 | -2 |
| 41.8 | 71.5 | 166 | 9.7279 | 15 | -2 |
| 40.877 | 84 | 153.8 | 8.7039 | 12 | -2 |
| 40.36 | 93.2 | 143.97 | 7.856 | 11 | -2 |
| 39.65 | 109.8 | 129.86 | 6.992 | 10 | -2 |
| 39.07 | 121.4 | 120.15 | 6.1439 | 9 | -2 |
| 38.65 | 130.83 | 111.3 | 6.832 | 8 | -2 |
| 38 | 144.4 | 99.33 | 5.968 | 7 | -2 |
| 37.63 | 153 | 91 | 5.1199 | 6 | -2 |
| 37.325 | 161.243 | 83 | 4.272 | 5 | -2 |
| 36.2 | 174.25 | 72 | 3.4 | 4 | -2 |
| 33.5 | 193 | 53.87 | 2.56 | 3 | -2 |
| 31.8 | 211.8 | 39.45 | 1.6959 | 2 | -2 |
| 30.88 | 224.66 | 30.1 | 0.848 | 1 | -2 |
| 27.5 | 16.45 | 250.83 | 295.93 | 505 | 2 |
| 23 | 12.56 | 252.8 | 390.01 | 671 | 2 |
| 26.945 | 16.363 | 251.1 | 311.12 | 539 | 2 |
| 22.6 | 11.7 | 253.92 | 356.04 | 606 | 2 |
| 24 | 13.4 | 254.53 | 322.89 | 560 | 2 |
| 30.8 | 26.36 | 248.25 | 193.13 | 327 | 2 |
| 25.5 | 16.4 | 253.9 | 264.17 | 448 | 2 |
| 30.8778 | 26.36 | 248.2 | 193.13 | 327 | 2 |
| 31.32 | 26.6 | 247.6 | 167 | 282 | 2 |
| 32.57 | 29.4 | 245.3 | 145.66 | 248 | 2 |
| 34.6 | 34.8 | 240.47 | 103.13 | 173 | 2 |
| 37 | 39.96 | 232.15 | 72.4 | 119 | 2 |
| 38.76 | 43.075 | 227 | 49.344 | 83 | 2 |
| 37.95 | 41.27 | 229.7 | 60.8 | 100 | 2 |
| 40 | 44.7 | 224.4 | 35.7 | 58 | 2 |
| 4.44 | 71.54 | 182.5 | 6.992 | 10 | 2 |
| 38.7 | 121.7 | 121 | 4.272 | 5 | 2 |
| 35.2 | 155.5 | 93.72 | 1.6959 | 2 | 2 |
| 34.57 | 193.75 | 49.9 | 0.848 | 1 | 2 |

- **Tabel Interpolasi Newton**

| i | xi | y1 | ST-1 | ST-2 | ST-3 | ST-4 | ST-5 | ST-6 | ST-7 | ST-8 | ST-9 | ST-10 | ST-11 | ST-12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | -12.000 | 67.000 | 2.000 | 1.625 | -0.188 | 0.013 | 0.000 | 0.000 | 0.000 | -0.00000106569 | 0.00000006459 | 0.00000000000 | -0.00000000055 | 0.00000000008 |
| 1 | -10.000 | 71.000 | 8.500 | 0.500 | -0.083 | 0.010 | -0.001 | 0.000 | 0.000 | 0.00000009688 | 0.00000006459 | -0.00000001211 | 0.00000000131 | |
| 2 | -8.000 | 88.000 | 10.500 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.00000125946 | -0.00000017762 | 0.00000001669 | | |
| 3 | -6.000 | 109.000 | 10.500 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | -0.00000193762 | 0.00000015609 | | | |
| 4 | -4.000 | 130.000 | 10.500 | 0.000 | 0.000 | 0.000 | -0.001 | 0.000 | 0.000 | 0.00000087193 | | | | |
| 5 | -2.000 | 151.000 | 10.500 | 0.000 | 0.000 | -0.008 | 0.001 | 0.000 | 0.000 | | | | | |
| 6 | 0.000 | 172.000 | 10.500 | 0.000 | -0.063 | 0.003 | 0.000 | 0.000 | | | | | | |
| 7 | 2.000 | 193.000 | 10.500 | -0.375 | -0.042 | 0.005 | -0.001 | | | | | | | |
| 8 | 4.000 | 214.000 | 9.000 | -0.625 | 0.000 | -0.003 | | | | | | | | |
| 9 | 6.000 | 232.000 | 6.500 | -0.625 | -0.021 | | | | | | | | | |
| 10 | 8.000 | 245.000 | 4.000 | -0.750 | | | | | | | | | | |
| 11 | 10.000 | 253.000 | 1.000 | | | | | | | | | | | |
| 12 | 12.000 | 255.000 | | | | | | | | | | | | |

| P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 | P9 | P10 | P11 | P12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 63.000 | 76.000 | 85.000 | 90.000 | 91.000 | 88.000 | 81.000 | 70.000 | 58.000 | 58.000 | 103.000 | 255.000 |
| 64.000 | 72.531 | 77.945 | 80.765 | 81.301 | 79.761 | 76.295 | 71.019 | 65.423 | 65.423 | 85.418 | 151.547 |
| 65.000 | 69.875 | 72.688 | 74.055 | 74.301 | 73.624 | 72.158 | 69.998 | 67.772 | 67.772 | 75.340 | 99.840 |
| 66.000 | 68.031 | 69.086 | 69.562 | 69.643 | 69.430 | 68.988 | 68.358 | 67.727 | 67.727 | 69.765 | 76.216 |
| 67.000 | 67.000 | 67.000 | 67.000 | 67.000 | 67.000 | 67.000 | 67.000 | 67.000 | 67.000 | 67.000 | 67.000 |
| 68.000 | 66.781 | 66.289 | 66.101 | 66.073 | 66.140 | 66.268 | 66.438 | 66.598 | 66.598 | 66.133 | 64.726 |
| 69.000 | 67.375 | 66.813 | 66.617 | 66.590 | 66.651 | 66.764 | 66.908 | 67.039 | 67.039 | 66.679 | 65.613 |
| 70.000 | 68.781 | 68.430 | 68.320 | 68.306 | 68.336 | 68.389 | 68.454 | 68.512 | 68.512 | 68.362 | 67.932 |
| 71.000 | 71.000 | 71.000 | 71.000 | 71.000 | 71.000 | 71.000 | 71.000 | 71.000 | 71.000 | 71.000 | 71.000 |
| 72.000 | 74.031 | 74.383 | 74.468 | 74.478 | 74.460 | 74.432 | 74.401 | 74.375 | 74.375 | 74.434 | 74.598 |
| 73.000 | 77.875 | 78.438 | 78.555 | 78.566 | 78.546 | 78.515 | 78.482 | 78.456 | 78.456 | 78.513 | 78.665 |
| 74.000 | 82.531 | 83.023 | 83.109 | 83.117 | 83.104 | 83.086 | 83.068 | 83.054 | 83.054 | 83.082 | 83.156 |
| 75.000 | 88.000 | 88.000 | 88.000 | 88.000 | 88.000 | 88.000 | 88.000 | 88.000 | 88.000 | 88.000 | 88.000 |
| 76.000 | 94.281 | 93.227 | 93.117 | 93.109 | 93.120 | 93.133 | 93.145 | 93.154 | 93.154 | 93.138 | 93.101 |
| 77.000 | 101.375 | 98.563 | 98.367 | 98.355 | 98.370 | 98.387 | 98.402 | 98.412 | 98.412 | 98.396 | 98.356 |
| 78.000 | 109.281 | 103.867 | 103.679 | 103.670 | 103.680 | 103.692 | 103.701 | 103.707 | 103.707 | 103.698 | 103.676 |
| 79.000 | 118.000 | 109.000 | 109.000 | 109.000 | 109.000 | 109.000 | 109.000 | 109.000 | 109.000 | 109.000 | 109.000 |
| 80.000 | 127.531 | 113.820 | 114.296 | 114.311 | 114.298 | 114.287 | 114.278 | 114.273 | 114.273 | 114.280 | 114.294 |
| 81.000 | 137.875 | 118.188 | 119.555 | 119.582 | 119.562 | 119.544 | 119.533 | 119.526 | 119.526 | 119.534 | 119.551 |
| 82.000 | 149.031 | 121.961 | 124.781 | 124.809 | 124.791 | 124.778 | 124.770 | 124.765 | 124.765 | 124.770 | 124.780 |
| 83.000 | 161.000 | 125.000 | 130.000 | 130.000 | 130.000 | 130.000 | 130.000 | 130.000 | 130.000 | 130.000 | 130.000 |
| 84.000 | 173.781 | 127.164 | 135.257 | 135.176 | 135.207 | 135.224 | 135.234 | 135.238 | 135.238 | 135.234 | 135.227 |
| 85.000 | 187.375 | 128.313 | 140.617 | 140.371 | 140.433 | 140.463 | 140.478 | 140.485 | 140.485 | 140.479 | 140.470 |
| 86.000 | 201.781 | 128.305 | 146.164 | 145.628 | 145.695 | 145.723 | 145.735 | 145.740 | 145.740 | 145.736 | 145.730 |
| 87.000 | 217.000 | 127.000 | 152.000 | 151.000 | 151.000 | 151.000 | 151.000 | 151.000 | 151.000 | 151.000 | 151.000 |
| 88.000 | 233.031 | 124.258 | 158.250 | 156.550 | 156.337 | 156.284 | 156.266 | 156.260 | 156.260 | 156.264 | 156.270 |
| 89.000 | 249.875 | 119.938 | 165.055 | 162.348 | 161.671 | 161.558 | 161.525 | 161.515 | 161.515 | 161.520 | 161.529 |
| 90.000 | 267.531 | 113.898 | 172.578 | 168.470 | 166.930 | 166.801 | 166.770 | 166.761 | 166.761 | 166.765 | 166.771 |
| 91.000 | 286.000 | 106.000 | 181.000 | 175.000 | 172.000 | 172.000 | 172.000 | 172.000 | 172.000 | 172.000 | 172.000 |
| 92.000 | 305.281 | 96.102 | 190.523 | 182.025 | 176.714 | 177.156 | 177.222 | 177.235 | 177.235 | 177.231 | 177.224 |
| 93.000 | 325.375 | 84.063 | 201.367 | 189.637 | 180.839 | 182.305 | 182.449 | 182.475 | 182.475 | 182.468 | 182.458 |
| 94.000 | 346.281 | 69.742 | 213.773 | 197.930 | 184.067 | 187.532 | 187.703 | 187.728 | 187.728 | 187.722 | 187.714 |
| 95.000 | 368.000 | 53.000 | 228.000 | 207.000 | 186.000 | 193.000 | 193.000 | 193.000 | 193.000 | 193.000 | 193.000 |
| 96.000 | 390.531 | 33.695 | 244.328 | 216.945 | 186.140 | 198.976 | 198.346 | 198.288 | 198.288 | 198.298 | 198.308 |
| 97.000 | 413.875 | 11.688 | 263.055 | 227.863 | 183.874 | 205.869 | 203.708 | 203.578 | 203.578 | 203.594 | 203.611 |
| 98.000 | 438.031 | -13.164 | 284.500 | 239.850 | 178.457 | 214.270 | 208.994 | 208.834 | 208.834 | 208.849 | 208.863 |
| 99.000 | 463.000 | -41.000 | 309.000 | 253.000 | 169.000 | 225.000 | 214.000 | 214.000 | 214.000 | 214.000 | 214.000 |
| 100.000 | 488.781 | -71.961 | 336.914 | 267.405 | 154.453 | 239.167 | 218.367 | 218.997 | 218.997 | 218.969 | 218.947 |
| 101.000 | 515.375 | -106.188 | 368.617 | 283.152 | 133.589 | 258.225 | 221.502 | 223.728 | 223.728 | 223.671 | 223.631 |
| 102.000 | 542.781 | -143.820 | 404.507 | 300.325 | 104.983 | 284.047 | 222.494 | 228.089 | 228.089 | 228.030 | 227.992 |
| 103.000 | 571.000 | -185.000 | 445.000 | 319.000 | 67.000 | 319.000 | 220.000 | 232.000 | 232.000 | 232.000 | 232.000 |
| 104.000 | 600.031 | -229.867 | 490.531 | 339.247 | 17.770 | 366.037 | 212.115 | 235.437 | 235.437 | 235.586 | 235.659 |
| 105.000 | 629.875 | -278.563 | 541.555 | 361.129 | -44.829 | 428.789 | 196.208 | 238.496 | 238.496 | 238.856 | 239.008 |
| 106.000 | 660.531 | -331.227 | 598.546 | 384.699 | -123.190 | 511.671 | 168.733 | 241.477 | 241.477 | 241.942 | 242.106 |
| 107.000 | 692.000 | -388.000 | 662.000 | 410.000 | -220.000 | 620.000 | 125.000 | 245.000 | 245.000 | 245.000 | 245.000 |
| 108.000 | 724.281 | -449.023 | 732.429 | 437.066 | -338.262 | 760.119 | 58.920 | 250.157 | 250.157 | 248.119 | 247.689 |
| 109.000 | 757.375 | -514.438 | 810.367 | 465.918 | -481.317 | 939.536 | -37.301 | 258.710 | 258.710 | 251.142 | 250.077 |
| 110.000 | 791.281 | -584.383 | 896.367 | 496.564 | -652.867 | 1167.066 | -173.510 | 273.349 | 273.349 | 253.354 | 251.947 |
| 111.000 | 826.000 | -659.000 | 991.000 | 529.000 | -857.000 | 1453.000 | -362.000 | 298.000 | 298.000 | 253.000 | 253.000 |

- **Pengambilan Data Berdasarkan Waktu**

| No. | Wkt | Lum1 | Exp | WB | Objek 1 (Merah) | | | | | | | | | Objek 2 (Biru) | | | | | | | | | Objek 3 (Hijau) | | | | | | | | | Objek 4 Kuning) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Hmin | Hmid | Hmax | Smin | Smid | Smax | Vmin | Vmid | Vmax | Hmin | Hmid | Hmax | Smin | Smid | Smax | Vmin | Vmid | Vmax | Hmin | Hmid | Hmax | Smin | Smid | Smax | Vmin | Vmid | Vmax | Hmin | Hmid | Hmax | Smin | Smid | Smax | Vmin | Vmid | Vmax |
| 1 | 9:00 | 32767 | -10 | DLA | 0 | 3 | 211 | 0 | 239 | 115 | 134 | 154 | 148 | 150 | 161 | 247 | 255 | 255 | 43 | 61 | 80 | 100 | 104 | 109 | 243 | 255 | 258 | 59 | 77 | 96 | 38 | 44 | 50 | 201 | 218 | 235 | 101 | 115 | 130 |
| 2 | | 32767 | -5 | DLA | 0 | 1 | 3 | 190 | 0 | 215 | 168 | 180 | 192 | 146 | 149 | 153 | 238 | 246 | 255 | 65 | 79 | 94 | 101 | 107 | 113 | 247 | 252 | 257 | 86 | 98 | 111 | 43 | 48 | 53 | 170 | 189 | 209 | 137 | 159 | 181 |
| 3 | | 32767 | 0 | DLA | 0 | 2 | 4 | 175 | 0 | 194 | 201 | 220 | 240 | 143 | 147 | 152 | 212 | 229 | 247 | 117 | 139 | 161 | 104 | 109 | 115 | 219 | 237 | 255 | 124 | 165 | 152 | 39 | 44 | 49 | 146 | 170 | 194 | 190 | 205 | 220 |
| 4 | | 32767 | 5 | DLA | 0 | 3 | 6 | 136 | 0 | 171 | 249 | 253 | 257 | 140 | 145 | 151 | 181 | 198 | 215 | 166 | 184 | 202 | 106 | 110 | 115 | 177 | 206 | 235 | 180 | 212 | 244 | 42 | 44 | 46 | 58 | 135 | 115 | 250 | 253 | 257 |
| 5 | | 32767 | 10 | DLA | 0 | 4 | 9 | 77 | 0 | 135 | 233 | 255 | 255 | 136 | 142 | 149 | 151 | 171 | 191 | 206 | 226 | 246 | 109 | 113 | 118 | 154 | 175 | 197 | 234 | 247 | 260 | 42 | 45 | 48 | 58 | 86 | 115 | 250 | 255 | 257 |
| 6 | 11:00 | 32767 | -10 | DLA | 0 | 3 | 9 | 206 | 219 | 229 | 130 | 139 | 155 | 146 | 153 | 160 | 246 | 251 | 256 | 43 | 58 | 73 | 99 | 107 | 112 | 245 | 255 | 260 | 60 | 82 | 85 | 42 | 46 | 50 | 184 | 205 | 215 | 110 | 113 | 145 |
| 7 | | 32767 | -5 | DLA | 0 | 3 | 6 | 184 | 197 | 202 | 182 | 189 | 203 | 147 | 151 | 156 | 215 | 236 | 257 | 71 | 86 | 102 | 106 | 109 | 112 | 222 | 241 | 256 | 103 | 115 | 131 | 42 | 44 | 51 | 178 | 185 | 197 | 151 | 168 | 187 |
| 8 | | 32767 | 0 | DLA | 0 | 3 | 9 | 161 | 180 | 193 | 213 | 220 | 236 | 146 | 150 | 154 | 178 | 207 | 237 | 111 | 133 | 158 | 107 | 111 | 115 | 194 | 223 | 249 | 111 | 167 | 211 | 41 | 44 | 52 | 154 | 170 | 174 | 205 | 208 | 221 |
| 9 | | 32767 | 5 | DLA | 0 | 4 | 6 | 125 | 140 | 156 | 250 | 255 | 259 | 143 | 148 | 154 | 158 | 178 | 198 | 160 | 184 | 208 | 106 | 112 | 119 | 171 | 192 | 221 | 168 | 211 | 255 | 42 | 45 | 48 | 127 | 132 | 145 | 245 | 252 | 259 |
| 10 | | 32767 | 10 | DLA | 3 | 6 | 14 | 79 | 92 | 105 | 247 | 255 | 259 | 137 | 143 | 150 | 133 | 149 | 165 | 216 | 210 | 249 | 106 | 114 | 119 | 35 | 163 | 194 | 250 | 230 | 255 | 42 | 46 | 49 | 52 | 66 | 80 | 252 | 253 | 255 |
| 11 | | 32767 | -10 | DLS | 0 | 257 | 6 | 250 | 247 | 255 | 24 | 48 | 53 | 160 | 163 | 167 | 246 | 253 | 255 | 0 | 16 | 33 | 87 | 91 | 94 | 244 | 247 | 255 | 0 | 32 | 23 | 48 | 53 | 57 | 242 | 247 | 260 | 28 | 46 | 60 |
| 12 | | 32767 | -5 | DLS | 249 | 252 | 255 | 244 | 247 | 257 | 49 | 61 | 96 | 153 | 157 | 161 | 249 | 252 | 256 | 61 | 75 | 90 | 94 | 101 | 108 | 249 | 247 | 258 | 27 | 44 | 49 | 46 | 49 | 61 | 207 | 223 | 248 | 57 | 98 | 98 |
| 13 | | 32767 | 0 | DLS | 244 | 252 | 256 | 216 | 221 | 237 | 75 | 102 | 118 | 152 | 154 | 157 | 244 | 251 | 259 | 71 | 88 | 105 | 104 | 109 | 116 | 245 | 247 | 258 | 48 | 60 | 83 | 48 | 52 | 58 | 179 | 194 | 210 | 93 | 125 | 160 |
| 14 | | 32767 | 5 | DLS | 248 | 251 | 255 | 184 | 196 | 220 | 129 | 155 | 183 | 146 | 150 | 154 | 238 | 247 | 257 | 108 | 122 | 136 | 107 | 112 | 117 | 240 | 247 | 255 | 47 | 97 | 147 | 52 | 54 | 56 | 157 | 168 | 179 | 158 | 175 | 192 |
| 15 | | 32767 | 10 | DLS | 248 | 251 | 258 | 165 | 173 | 182 | 190 | 207 | 218 | 144 | 147 | 150 | 213 | 232 | 246 | 160 | 174 | 188 | 111 | 116 | 120 | 249 | 247 | 258 | 136 | 147 | 167 | 47 | 50 | 54 | 125 | 139 | 154 | 212 | 226 | 240 |
| 16 | 15:30 | 6754 | -10 | DLA | | | | | | | | | | 158 | 161 | 171 | 250 | 247 | 255 | 47 | 67 | 79 | | | | | | | | | | | | | | | | | | |
| 17 | | 6520 | -5 | DLA | | | | | | | | | | 155 | 155 | 161 | 250 | 247 | 256 | 72 | 97 | 108 | | | | | | | | | | | | | | | | | | |
| 18 | | 6268 | 0 | DLA | | | | | | | | | | 151 | 153 | 156 | 237 | 244 | 257 | 114 | 141 | 164 | | | | | | | | | | | | | | | | | | |
| 19 | | 6063 | 5 | DLA | | | | | | | | | | 150 | 151 | 155 | 213 | 226 | 246 | 174 | 198 | 218 | | | | | | | | | | | | | | | | | | |
| 20 | | 5923 | 10 | DLA | | | | | | | | | | 144 | 148 | 153 | 192 | 202 | 217 | 209 | 224 | 260 | | | | | | | | | | | | | | | | | | |
| 21 | | 5116 | -10 | DLA | 242 | 248 | 254 | 212 | 231 | 251 | 79 | 108 | 120 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 22 | | 5194 | -5 | DLA | 246 | 249 | 251 | 188 | 202 | 218 | 121 | 154 | 173 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 23 | | 5290 | 0 | DLA | 244 | 248 | 250 | 175 | 182 | 198 | 172 | 198 | 206 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 24 | | 5448 | 5 | DLA | 245 | 248 | 248 | 143 | 155 | 170 | 215 | 225 | 249 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 25 | | 5923 | 10 | DLA | 238 | 243 | 247 | 94 | 111 | 144 | 246 | 230 | 258 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 26 | | 5485 | -10 | | | | | | | | | | | | | | | | | | | | 107 | 114 | 119 | 246 | 247 | 258 | 49 | 64 | 74 | | | | | | | | | |
| 27 | | 5673 | -5 | | | | | | | | | | | | | | | | | | | | 111 | 116 | 121 | 240 | 247 | 255 | 46 | 96 | 148 | | | | | | | | | |
| 28 | | 5680 | 0 | | | | | | | | | | | | | | | | | | | | 115 | 118 | 122 | 249 | 247 | 256 | 117 | 133 | 145 | | | | | | | | | |
| 29 | | 5756 | 5 | | | | | | | | | | | | | | | | | | | | 119 | 120 | 126 | 248 | 247 | 257 | 170 | 194 | 209 | | | | | | | | | |
| 30 | | 5900 | 10 | | | | | | | | | | | | | | | | | | | | 121 | 123 | 126 | 233 | 247 | 259 | 224 | 223 | 246 | | | | | | | | | |
| 31 | | 5600 | -10 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 54 | 56 | 61 | 153 | 170 | 185 | 86 | 107 | 136 |
| 32 | | 5742 | -5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 53 | 56 | 58 | 133 | 148 | 166 | 137 | 159 | 189 |
| 33 | | 5750 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 49 | 54 | 59 | 123 | 133 | 146 | 173 | 196 | 204 |
| 34 | | 5874 | 5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 51 | 53 | 58 | 86 | 100 | 116 | 228 | 225 | 252 |
| 35 | | 5874 | 10 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 40 | 43 | 45 | 44 | 56 | 69 | 250 | 230 | 25 |

- **Koding MainActivity OpenCV Android (Java):**

```
package mbs.TA05;
import java.util.List;
import org.opencv.android.BaseLoaderCallback;
import org.opencv.android.LoaderCallbackInterface;
import org.opencv.android.OpenCVLoader;
import org.opencv.android.CameraBridgeViewBase.CvCameraViewFrame;
import org.opencv.android.CameraBridgeViewBase.CvCameraViewListener2;
import org.opencv.core.Core;
import org.opencv.core.CvType;
import org.opencv.core.Mat;
import org.opencv.core.MatOfPoint;
import org.opencv.core.MatOfPoint2f;
import org.opencv.core.Point;
import org.opencv.core.Rect;
import org.opencv.core.Scalar;
import org.opencv.core.Size;
import org.opencv.imgproc.Imgproc;
import android.annotation.SuppressLint;
import android.app.Activity;
import android.content.Intent;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.MotionEvent;
import android.view.View;
import android.view.View.OnTouchListener;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.SeekBar;
public class SeekBarVal extends Activity implements CvCameraViewListener2,
OnTouchListener {
   private static final String TAG = "SeekActivity";
   private static final int    VIEW_MODE_HSV11   = 0;
   private MenuItem     mItemHsv11;
   private int          mViewMode;
   private float nLux;
   private float nLuxAcc;
   private MisiView  mOpenCvCameraView;
   private Mat mRgba;
   private Mat mHSV;
   public static double Hmin1 = 0;
   public static double Hmid = 0;
   public static double Hmax1 = 0;
   public static double Smin1 = 0;
   public static double Smid = 0;
   public static double Smax1 = 0;
   public static double Vmin1 = 0;
   public static double Vmid = 0;
   public static double Vmax1 = 0;
   public static double Dilate1;
   public static double Erode1;
   public static int ExVal = -12;
   public static int WBidx = 2;
   public static double valVarY = 0;
   public static double satVar = 0;
   public static int ExpVal = 12;
   public static double HueAv = 0;
   public static double SatAv = 0;
```

```
    public static double ValAv = 0;
    public static double CamL = 0;
    public static String valWB = "daylight";
    private Boolean val = false;
    private Sensor mOrienta;
    private SeekBar seekHmin1;
    private SeekBar seekHmax1;
    private SeekBar seekSmin1;
    private SeekBar seekSmax1;
    private SeekBar seekVmin1;
    private SeekBar seekVmax1;
    private SeekBar seekErode;
    private SeekBar seekDilate;
    private SeekBar seekEV;
    private SeekBar seekWB;
    private boolean mIsDisplayTouched;
    private SeekAdapter sH;
    private SeekAdapter sS;
    private SeekAdapter sV;
    private SeekAdapter sED;
    private SeekAdapter sEW;
    private Mat mSpectrum;
    private Scalar mBlobColorRgba;
    private Scalar mBlobColorHsv;
    private Scalar mBlobColorRgba2;
    private Scalar mBlobColorHsv2;
    private Size SPECTRUM_SIZE;
    private BlobDetector mDetector;
    protected boolean bIsHPressed;
    protected boolean bIsSPressed;
    protected boolean bIsVPressed;
    protected boolean bIsEDPressed;
    protected boolean bIsEWPressed;
    private SensorEventListener mySensorEventListener;
    private Mat mDisplay;
    private Mat mBiner;
    private boolean idxBiner =false;
    private BaseLoaderCallback mLoaderCallback = new BaseLoaderCallback(this) {
    @Override
    public void onManagerConnected(int status) {
    switch (status) {
    case LoaderCallbackInterface.SUCCESS:{
    Log.i(TAG, "OpenCV loaded successfully");
    mOpenCvCameraView.enableView();
    mOpenCvCameraView.setOnTouchListener(SeekBarVal.this);
} break;
default:
     {
       super.onManagerConnected(status);
     } break;
    }
 }
};
private MenuItem mItemExposure;
private Scalar CONTOUR_COLOR;
    public SeekBarVal() {
          Log.i(TAG, "Instantiated new " + this.getClass());
       }
@SuppressWarnings("deprecation")
@Override
public void onCreate(Bundle savedInstanceState) {
 Log.i(TAG, "called onCreate");
super.onCreate(savedInstanceState);
getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
```

```
setContentView(R.layout.biner);
SensorManager mySensorManager = (SensorManager)getSystemService(SENSOR_SERVICE);
Sensor LightSensor = mySensorManager.getDefaultSensor(Sensor.TYPE_LIGHT);
mySensorManager.registerListener(LightSensorListener, LightSensor,
SensorManager.SENSOR_DELAY_NORMAL);
mOpenCvCameraView = (MisiView) findViewById(R.id.activity_surface_view);
mOpenCvCameraView.setCameraIndex(0);
mOpenCvCameraView.setMaxFrameSize(480, 320);
mOpenCvCameraView.setCvCameraViewListener(this);
}
@Override
public boolean onCreateOptionsMenu(Menu menu) {
Log.i(TAG, "called onCreateOptionsMenu");
mItemExposure = menu.add("exp");
return true;
}
@Override
public void onPause()
{
super.onPause();
if (mOpenCvCameraView != null)
mOpenCvCameraView.disableView();
}
@Override
public void onResume()
{
super.onResume();
OpenCVLoader.initAsync(OpenCVLoader.OPENCV_VERSION_3_0_0, this,
mLoaderCallback);
}
@Override
protected void onDestroy(){
if (mOpenCvCameraView != null)
mOpenCvCameraView.disableView();
super.onDestroy();
}
@Override
public void onCameraViewStarted(int width, int height) {
// TODO Auto-generated method stub
mRgba = new Mat(height, width, CvType.CV_8UC4);
mHSV = new Mat(height, width, CvType.CV_8UC4);
mDetector = new BlobDetector();
sH = new SeekAdapter();
sS = new SeekAdapter();
sV = new SeekAdapter();
sED = new SeekAdapter();
sEW = new SeekAdapter();
mSpectrum = new Mat();
mBlobColorRgba = new Scalar(255);
mBlobColorHsv = new Scalar(255);
mBlobColorRgba2 = new Scalar(255);
mBlobColorHsv2 = new Scalar(255);
SPECTRUM_SIZE = new Size(mRgba.cols()/1.5, 64);
CONTOUR_COLOR = new Scalar(255,0,0,255);
mDisplay = new Mat();
mBiner = new Mat(height,width,CvType.CV_8UC1);
}

@Override
public void onCameraViewStopped() {
// TODO Auto-generated method stub
mRgba.release();
mHSV.release();
mDisplay.release();
```

```
mBiner.release();
}
@Override
public Mat onCameraFrame(CvCameraViewFrame inputFrame) {
final int viewMode = mViewMode;
mRgba = inputFrame.rgba();
mOpenCvCameraView.setFocus("manual");
mOpenCvCameraView.setWhite();
mOpenCvCameraView.setAntibanding("auto");
mOpenCvCameraView.setExposureLock(val);
Button bMenu = (Button)findViewById(R.id.tombolMenu);
bMenu.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v) {
    openOptionsMenu();
    }
});
Rect totScreen = new Rect();
    totScreen.x = 0;
    totScreen.y = 0;
    totScreen.width = 480;
    totScreen.height = 320;
    Mat touchedRegionRgba2 = mRgba.submat(totScreen);
    Mat touchedRegionHsv2 = new Mat();
    Imgproc.cvtColor(touchedRegionRgba2, touchedRegionHsv2,
Imgproc.COLOR_RGB2HSV_FULL);
    mBlobColorHsv2 = Core.sumElems(touchedRegionHsv2);
    HueAv = mBlobColorHsv2.val[0]/153600;
    SatAv = mBlobColorHsv2.val[1]/153600;
    ValAv = mBlobColorHsv2.val[2]/153600;
if (viewMode==VIEW_MODE_HSV11){
    Button bH = (Button)findViewById(R.id.tombolH);
    Button bS = (Button)findViewById(R.id.tombolS);
    Button bV = (Button)findViewById(R.id.tombolV);
    Button bED = (Button)findViewById(R.id.tombolED);
    Button bEW = (Button)findViewById(R.id.tombolEW);
    seekHmin1=(SeekBar) findViewById(R.id.min);
    seekHmax1=(SeekBar) findViewById(R.id.max);
    seekSmin1=(SeekBar) findViewById(R.id.min);
    seekSmax1=(SeekBar) findViewById(R.id.max);
    seekVmin1=(SeekBar) findViewById(R.id.min);
    seekVmax1=(SeekBar) findViewById(R.id.max);
    seekErode=(SeekBar) findViewById(R.id.min);
    seekDilate=(SeekBar) findViewById(R.id.max);
    seekEV=(SeekBar) findViewById(R.id.min);
    seekWB=(SeekBar) findViewById(R.id.max);

    bH.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v) {
    sH.setProgress(seekHmin1, seekHmax1, Hmin1, Hmax1, 262);
    sH.seekProgress();
    bIsHPressed = true;
    bIsSPressed = false;
    bIsVPressed = false;
    bIsEDPressed = false;
    bIsEWPressed = false;
    mIsDisplayTouched = false;
        }
    });
    bS.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v) {
    sS.setProgress(seekSmin1, seekSmax1, Smin1, Smax1, 262);
```

```
sS.seekProgress();
bIsSPressed = true;
bIsHPressed = false;
bIsVPressed = false;
bIsEDPressed = false;
bIsEWPressed = false;
mIsDisplayTouched = false;
}
});
bV.setOnClickListener(new View.OnClickListener(){
@Override
public void onClick(View v) {
sV.setProgress(seekVmin1, seekVmax1, Vmin1, Vmax1, 262);
sV.seekProgress();
sV.seekBarChange();
bIsHPressed = false;
bIsSPressed = false;
bIsEDPressed = false;
bIsEWPressed = false;
bIsVPressed = true;
mIsDisplayTouched = false;
}
});
bED.setOnClickListener(new View.OnClickListener(){
@Override
public void onClick(View v) {
sED.setProgress(seekErode, seekDilate, Erode1, Dilate1, 21);
sED.seekProgress();
sED.seekBarChange();
bIsEDPressed = true;
bIsHPressed = false;
bIsSPressed = false;
bIsVPressed = false;
bIsEWPressed = false;
mIsDisplayTouched = false;
if(idxBiner)
idxBiner = false;
else
idxBiner = true;
}
});
bEW.setOnClickListener(new View.OnClickListener(){
@Override
public void onClick(View v) {
sEW.setProgress(seekEV, seekWB, ExpVal, WBidx, 25);
sEW.seekProgress();
sEW.seekBarChange();
bIsEDPressed = false;
bIsHPressed = false;
bIsSPressed = false;
bIsVPressed = false;
bIsEWPressed = true;
mIsDisplayTouched = false;
}
});
if(bIsHPressed){
Hmin1 = sH.getFirstVal();
Hmax1 = sH.getSecondVal();
sH.setProgress(seekHmin1, seekHmax1, Hmin1, Hmax1, 262);
sH.seekProgress();
sH.seekBarChange();
}
if(bIsSPressed){
Smin1 = sS.getFirstVal();
```

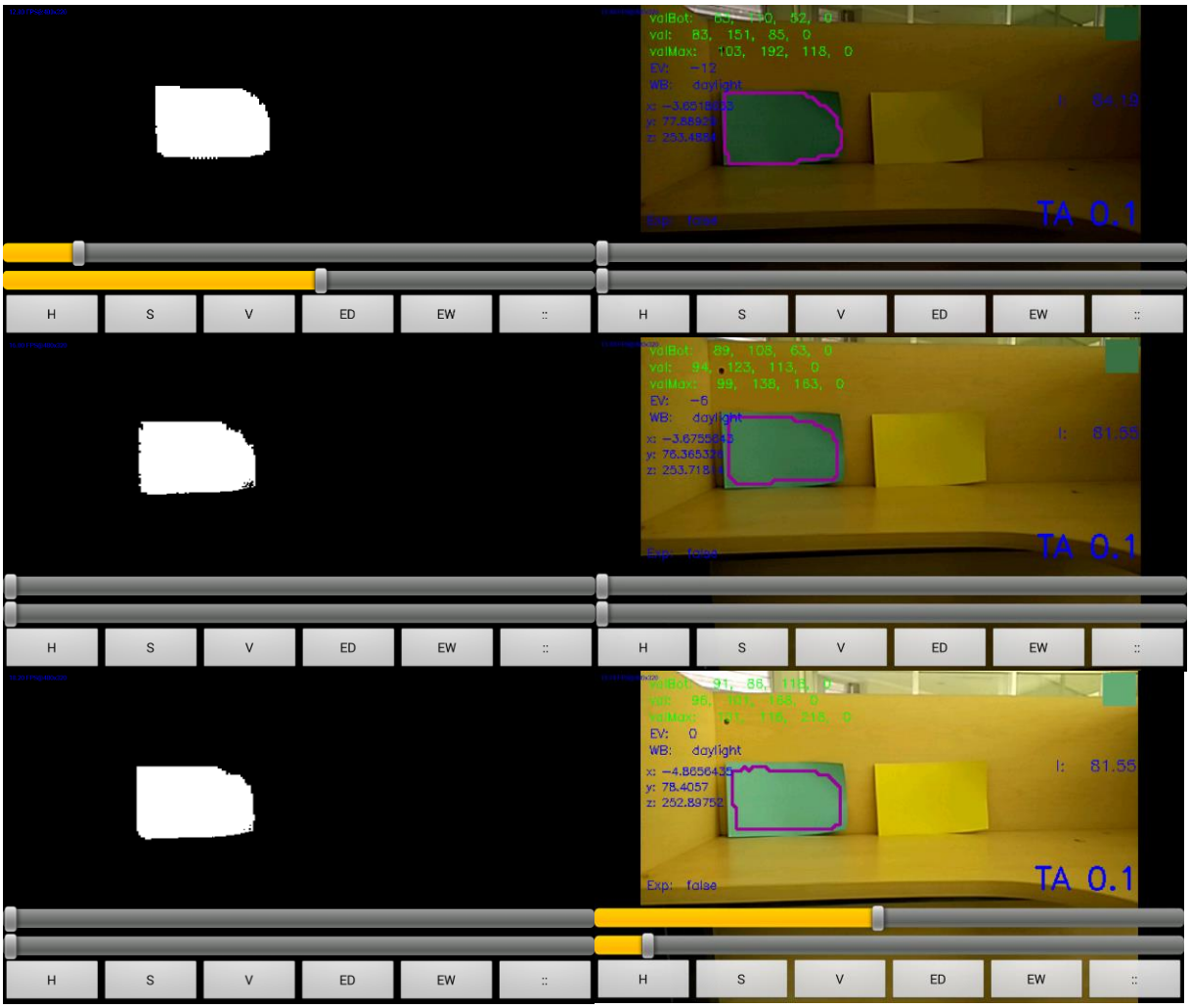
```
Smax1 = sS.getSecondVal();
sS.setProgress(seekSmin1, seekSmax1, Smin1, Smax1, 262);
sS.seekProgress();
sS.seekBarChange();
}
if(bIsVPressed){
Vmin1 = sV.getFirstVal();
Vmax1 = sV.getSecondVal();
sV.setProgress(seekVmin1, seekVmax1, Vmin1, Vmax1, 262);
sV.seekProgress();
sV.seekBarChange();
}
if(bIsEDPressed){
Erode1 = sED.getFirstVal();
Dilate1 = sED.getSecondVal();
sED.setProgress(seekErode, seekDilate, Erode1, Dilate1, 21);
sED.seekProgress();
sED.seekBarChange();
}
if(bIsEWPressed){
ExpVal = sEW.getFirstVal();
WBidx = sEW.getSecondVal();
sEW.setProgress(seekEV, seekWB, ExpVal, WBidx, 25);
sEW.seekProgress();
sEW.seekBarChange();
}
if(idxBiner){
Scalar hsv_min2 = new Scalar(Hmin1, Smin1, Vmin1, 0);
Scalar hsv_max2 = new Scalar(Hmax1, Smax1, Vmax1, 0);
Imgproc.cvtColor(mRgba, mHSV, Imgproc.COLOR_RGB2HSV_FULL,4);
Core.inRange(mHSV, hsv_min2, hsv_max2, mBiner);
Imgproc.erode(mBiner, mBiner,
    Imgproc.getStructuringElement(Imgproc.MORPH_ELLIPSE, new Size
    (2*Erode1+1,2*Erode1+1), new Point (Erode1,Erode1)));
Imgproc.dilate(mBiner, mBiner,
    Imgproc.getStructuringElement(Imgproc.MORPH_ELLIPSE, new Size
    (2*Dilate1+1,2*Dilate1+1), new Point (Dilate1,Dilate1)));
Imgproc.dilate(mBiner, mBiner,
    Imgproc.getStructuringElement(Imgproc.MORPH_ELLIPSE, new Size
    (2*Dilate1+1,2*Dilate1+1), new Point (Dilate1,Dilate1)));
Imgproc.erode(mBiner, mBiner,
    Imgproc.getStructuringElement(Imgproc.MORPH_ELLIPSE, new Size
    (2*Erode1+1,2*Erode1+1), new Point (Erode1,Erode1)));
mDisplay = mBiner;
}
else{
if(mIsDisplayTouched){
Hmin1 = (int) mDetector.getHmin();
Hmid  = (int) mDetector.getHmid();
Hmax1 = (int) mDetector.getHmax();
Smin1 = (int) mDetector.getSmin();
Smid  = (int) mDetector.getSmid();
Smax1 = (int) mDetector.getSmax();
Vmin1 = (int) mDetector.getVmin();
Vmid  = (int) mDetector.getVmid();
Vmax1 = (int) mDetector.getVmax();
}
mDetector.process(mRgba, new Scalar(Hmin1, Smin1, Vmin1), new Scalar(Hmax1,
    Smax1, Vmax1), Dilate1, Erode1);
List<MatOfPoint> contours = mDetector.getContours();
MatOfPoint approx =  new MatOfPoint();
double max = 0;
for (int i = 0; i < contours.size(); i++){
    MatOfPoint tempContour = contours.get(i);
```

```
        MatOfPoint2f newMat = new MatOfPoint2f( tempContour.toArray() );
        MatOfPoint2f newApprox = new MatOfPoint2f( approx.toArray() );

        Imgproc.approxPolyDP(newMat, newApprox, Imgproc.arcLength(newMat,
true)*0.02, true);

    if (Imgproc.contourArea(contours.get(i)) > max){
       max = Imgproc.contourArea(contours.get(i));
    }
    }
    switch(WBidx){
    case 0  : valWB = "auto"; break;
    case 1  : valWB = "auto"; break;
    case 2  : valWB = "daylight"; break;
    case 3  : valWB = "daylight"; break;
    case 4  : valWB = "cloudy-daylight"; break;
    case 5  : valWB = "cloudy-daylight"; break;
    case 6  : valWB = "twilight"; break;
    case 7  : valWB = "twilight"; break;
    case 8  : valWB = "incandescent"; break;
    case 9  : valWB = "incandescent"; break;
    case 10 : valWB = "warm-fluorescent"; break;
    case 11 : valWB = "warm-fluorescent"; break;
    case 12 : valWB = "fluorescent"; break;
    case 13 : valWB = "fluorescent"; break;
    case 14 : valWB = "shade"; break;
    case 15 : valWB = "shade"; break;
    case 16 : valWB = "auto"; break;
    case 17 : valWB = "auto"; break;
    default : valWB = "daylight"; break;
    }
    ExVal = ExpVal-12;
    mOpenCvCameraView.setExposure(ExVal);
    Imgproc.drawContours(mRgba, contours, -1, new Scalar(153, 0, 153, 155), 2);
    Imgproc.putText(mRgba, "TA 0.1: " + max, new Point (350, 210),
        Core.FONT_HERSHEY_SIMPLEX, 0.4, new Scalar(0, 0, 255), 1);
    Imgproc.putText(mRgba, "valBot:   " + Hmin1 + ",  " + Smin1 + ",  " + Vmin1
        + ",  " + Erode1, new Point (8, mRgba.rows() * 0.05),
        Core.FONT_HERSHEY_SIMPLEX, 0.4, new Scalar(0, 255, 0), 1);
    Imgproc.putText(mRgba, "valM1:   " + Hmid + ",  " + Smid + ",  " + Vmid + ",
        " + Dilate1, new Point (8, mRgba.rows() * 0.1),
        Core.FONT_HERSHEY_SIMPLEX, 0.4, new Scalar(255, 0, 0), 1);
    Imgproc.putText(mRgba, "valM2:   " + ((Hmin1+Hmax1)/2) + ",  " +
        ((Smin1+Smax1)/2) + ",  " + ((Vmin1+Vmax1)/2) + ",  " + Dilate1, new
        Point (8, mRgba.rows() * 0.15), Core.FONT_HERSHEY_SIMPLEX, 0.4, new
        Scalar(255, 0, 0), 1);
    Imgproc.putText(mRgba, "valMax:   " + Hmax1 + ",  " + Smax1 + ",  " + Vmax1
        + ",  " + Dilate1, new Point (8, mRgba.rows() * 0.2),
        Core.FONT_HERSHEY_SIMPLEX, 0.4, new Scalar(0, 255, 0), 1);
    Imgproc.putText(mRgba, "I1: " + nLux, new Point ( 400, mRgba.rows() * 0.3),
        Core.FONT_HERSHEY_SIMPLEX, 0.5, new Scalar(0, 0, 255), 1);
    Imgproc.putText(mRgba, "I2: " + nLuxAcc, new Point ( 400, mRgba.rows() *
        0.5), Core.FONT_HERSHEY_SIMPLEX, 0.5, new Scalar(0, 0, 255), 1);
    Imgproc.putText(mRgba, "EV:   " + ExVal, new Point (8, mRgba.rows() * 0.25),
        Core.FONT_HERSHEY_SIMPLEX, 0.4, new Scalar(0, 0, 255), 1);
    Imgproc.putText(mRgba, "WB:   " + valVarY, new Point (8, mRgba.rows() *
        0.3), Core.FONT_HERSHEY_SIMPLEX, 0.4, new Scalar(0, 0, 255), 1);
    CamL = ((Math.pow(HueAv, 1.61)*(-1.35)+Math.pow(SatAv,
        0.985)*4.45+Math.pow(ValAv, 1.247)*1.413)/1.25)-440;
    CamL = this.expGrowthFunc(255-SatAv);
    Imgproc.putText(mRgba, "Exp:  " + val, new Point (5, 160),
        Core.FONT_HERSHEY_SIMPLEX, 0.4, new Scalar(0, 0, 255), 1);
    Imgproc.putText(mRgba, "HueVal: " + HueAv, new Point (5, 115),
        Core.FONT_HERSHEY_SIMPLEX, 0.35, new Scalar(0, 0, 255), 1);
```

```
Imgproc.putText(mRgba, "SatVal: " + SatAv, new Point (5, 130),
    Core.FONT_HERSHEY_SIMPLEX, 0.35, new Scalar(0, 0, 255), 1);
Imgproc.putText(mRgba, "AvVal : " + ValAv, new Point (5, 145),
    Core.FONT_HERSHEY_SIMPLEX, 0.35, new Scalar(0, 0, 255), 1);
Imgproc.putText(mRgba, "CamLux : " + CamL, new Point (5, 200),
    Core.FONT_HERSHEY_SIMPLEX, 0.5, new Scalar(0, 0, 255), 2);
Mat colorLabel = mRgba.submat(2, 34, 446, 478);
colorLabel.setTo(mBlobColorRgba);
mDisplay = mRgba;
}
}
return mDisplay;
}
public double expGrowthFunc (double x){
return (0.000005556*Math.exp(0.07802*x));
}
public boolean onOptionsItemSelected(MenuItem item) {
if (item == mItemExposure){
if(val)
val = false;
else
val = true;
}
return true;
}
@SuppressLint("ClickableViewAccessibility")
@Override
public boolean onTouch(View v, MotionEvent event) {
int cols = mRgba.cols();
int rows = mRgba.rows();
int x = (int)(event.getX() * 0.25);
int y = (int)(event.getY() * 0.296296296);
Log.i(TAG, "Touch image coordinates: (" + x + ", " + y + ")");
if ((x < 0) || (y < 0) || (x > cols) || (y > rows)) return false;
Rect touchedRect = new Rect();
touchedRect.x = (x>4) ? x-4 : 0;
touchedRect.y = (y>4) ? y-4 : 0;
touchedRect.width = (x+4 < cols) ? x + 4 - touchedRect.x : cols -
    touchedRect.x;
touchedRect.height = (y+4 < rows) ? y + 4 - touchedRect.y : rows -
    touchedRect.y;
Mat touchedRegionRgba = mRgba.submat(touchedRect);
Mat touchedRegionHsv = new Mat();
Imgproc.cvtColor(touchedRegionRgba, touchedRegionHsv,
    Imgproc.COLOR_RGB2HSV_FULL);
int pointCount = touchedRect.width*touchedRect.height;
mBlobColorHsv = Core.sumElems(touchedRegionHsv);
for (int i = 0; i < mBlobColorHsv.val.length; i++)
mBlobColorHsv.val[i] /= pointCount;
mBlobColorRgba = converScalarHsv2Rgba(mBlobColorHsv);
Log.i(TAG, "Touched rgba color: (" + mBlobColorRgba.val[0] + ", " +
    mBlobColorRgba.val[1] +
     ", " + mBlobColorRgba.val[2] + ", " + mBlobColorRgba.val[3] + ")");
mDetector.setHsvColor(mBlobColorHsv);
Imgproc.resize(mDetector.getSpectrum(), mSpectrum, SPECTRUM_SIZE);
mDetector.process(mRgba);
bIsHPressed = false;
bIsSPressed = false;
bIsVPressed = false;
bIsEDPressed = false;
bIsEWPressed = false;
mIsDisplayTouched = true;
touchedRegionRgba.release();
touchedRegionHsv.release();
```

```
return false;
}
private Scalar converScalarHsv2Rgba(Scalar hsvColor) {
Mat pointMatRgba = new Mat();
Mat pointMatHsv = new Mat(1, 1, CvType.CV_8UC3, hsvColor);
Imgproc.cvtColor(pointMatHsv, pointMatRgba, Imgproc.COLOR_HSV2RGB_FULL, 4);
return new Scalar(pointMatRgba.get(0, 0));
}
private final SensorEventListener LightSensorListener = new
    SensorEventListener(){
@Override
public void onAccuracyChanged(Sensor sensor, int accuracy) {
nLuxAcc = accuracy;
}
@Override
public void onSensorChanged(SensorEvent event) {
nLux = event.values[0];
nLuxAcc = event.values[1];
}};
}
```

- **Dokumentasi**



- **Koding Simulasi Penerapan dari Gambar (C/C++).**

```
#include <iostream>
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include "opencv/highgui.h"
#include "opencv/cv.h"
using namespace cv;
using namespace std;
 int H_Min = 97;
 int H_Max = 103;
 int S_Min = 92;
 int S_Max = 255;
 int V_Min = 70;
 int V_Max = 255;
 int Erode = 0;
 int Dilate = 1;
 int GB1 = 0;
 int GB2 = 4;
 Mat imgHSV, imgAsli, imgBiner;
 int resX = 480;
 int resY = 320;
 char** tempImage;
 Point centerRef = Point(288,133);
 Point topLeftRef = Point(141,77);
 Point botLeftRef = Point(135,191);
 Point topRightRef = Point(439,74);
 Point botRightRef = Point(440,195);
 int areaRef    = 22346;
 Point centerDif = Point(0, 0);
 int trimRate = 0;
 int listRate = 0;
 int SkewRate = 0;
 vector<Point> pts2;
 int vertice;
int lengthPoint(Point centerA, Point centerB){
    return sqrt(pow(centerA.x-centerB.x,2)+pow(centerA.y-centerB.y,2));
}
vector<Point> contoursConvexHull( vector<vector<Point> > contours )
{
    vector<Point> result;
    vector<Point> pts;
    Point topRightEnd(0, 0);
    Point topLeftEnd(0, 0);
    Point botRightEnd(resX, resY);
    Point botLeftEnd(resX, resY);
    int topLengthL = 0;
    int topLengthR = 0;
    int botLengthL = 0;
    int botLengthR = 0;
    int topLeftCount = 0;
    int topRightCount = 0;
    int botLeftCount = 0;
    int botRightCount = 0;
    int area = 0;
    for ( size_t i = 0; i< contours.size(); i++){
    area = contourArea(contours[i]) + area;
    for ( size_t j = 0; j< contours[i].size(); j++){
    pts.push_back(contours[i][j]);
    topLengthL = sqrt(pow(pts[j].x,2)+pow(pts[j].y,2));
    botLengthR = sqrt(pow(pts[j].x,2)+pow(pts[j].y,2));
    topLengthR = sqrt(pow(resX-pts[j].x,2)+pow(pts[j].y,2));
    botLengthL = sqrt(pow(resX-pts[j].x,2)+pow(pts[j].y,2));
    if(pts[j].x < botRightEnd.x || pts[j].y < botRightEnd.y){
    botRightEnd.x = pts[j].x;
```

```
botRightEnd.y = pts[j].y;
topLeftCount = j;
}
if(pts[j].x > topLeftEnd.x && pts[j].y > topLeftEnd.y){
topLeftEnd.x = pts[j].x;
topLeftEnd.y = pts[j].y;
botRightCount = j;
}
if(topLengthR <           sqrt(pow(botLeftEnd.x,2)+pow(botLeftEnd.y,2))){
botLeftEnd.x = resX-pts[j].x;
botLeftEnd.y = pts[j].y;
topRightCount = j;
}
  if(botLengthL > sqrt(pow(topRightEnd.x,2)+pow(topRightEnd.y,2))){
  topRightEnd.x = resX-pts[j].x;
  topRightEnd.y = pts[j].y;
  botLeftCount = j;
  }
  }
  }
  circle (imgAsli, pts[topLeftCount], 4, Scalar( 0, 255, 0 ), 2, 3 ,0 );
  circle (imgAsli, pts[botRightCount], 4, Scalar( 0, 255, 0 ), 2, 3 ,0 );
  circle (imgAsli, pts[topRightCount], 4, Scalar( 0, 255,               0 ),
  2, 3 ,0 );
  circle (imgAsli, pts[botLeftCount], 4, Scalar( 0, 255,               0 ),
  2, 3 ,0 );
  line(imgAsli, pts[topLeftCount], pts[topRightCount],          Scalar(255,
  0, 0), 2);
  line(imgAsli, pts[topLeftCount], pts[botLeftCount],           Scalar(255,
  0, 0), 2);
  line(imgAsli, pts[botRightCount], pts[botLeftCount], Scalar(255, 0, 0), 2);
  line(imgAsli, pts[botRightCount], pts[topRightCount],Scalar(255, 0, 0), 2);
  int topL = lengthPoint(pts[topRightCount], pts[topLeftCount]);
  int rigL = lengthPoint(pts[topRightCount], pts[botRightCount]);
  int botL = lengthPoint(pts[botRightCount], pts[botLeftCount]);
  int lefL = lengthPoint(pts[topLeftCount], pts[botLeftCount]);
  Point center =
  Point(((pts[topRightCount].x+pts[topLeftCount].x)/2+(pts[botRightCount].x+p
  ts[botLeftCount].x)/2)/2, ((pts[topRightCount].y+pts[botRightCount].y)/2 +
  (pts[topLeftCount].y+pts[botLeftCount].y)/2)/2);
  Point centerUp = Point((pts[topRightCount].x+pts[topLeftCount].x)/2,
  (pts[topRightCount].y+pts[topLeftCount].y)/2);
  Point centerVer =
  Point(((pts[topRightCount].x+pts[topLeftCount].x)/2+(pts[botRightCount].x+p
  ts[botLeftCount].x)/2)/2, ((pts[topRightCount].y+pts[botRightCount].y)/2 +
  (pts[topLeftCount].y+pts[botLeftCount].y)/2)/2 - 55);
  line(imgAsli, center, centerUp, Scalar(0, 255, 255), 2);
  line(imgAsli, center, centerVer, Scalar(150, 0, 255), 2);
  circle (imgAsli, center, 4, Scalar( 0, 0, 255 ), 3, 3 ,0 );
  circle (imgAsli, centerRef, 4, Scalar( 0, 0, 255 ), 3, 3 ,0 );
  int lengthDif = lengthPoint(center, centerRef);
  line(imgAsli, center, centerRef, Scalar(0, 255, 255), 2);
  float listRef = (float)121/(float)114;
  float list = (float)rigL/(float)lefL;
  float listScale = 0.01/abs((float)121/(float)114-
  (float)121/((float)114+1));
  float listRate = (listRef-list)*listScale;
  float listRate = lineDifferene(Point(121, 114), Point(rigL, lefL));
  float trimRef = (float)305/(float)298;
  float trim = (float)botL/(float)topL;
  float trimScale = 0.01/abs((float)305/(float)298-
  (float)305/((float)298+(float)1));
  float trimRate = (trimRef-trim)*trimScale;
```

```
float skewRate = asin(((float)centerUp.x-
(float)center.x)/(float)lengthPoint(centerUp,center))*57.29578;
Point topPos = Point((pts[topRightCount].x+pts[topLeftCount].x)/2 - 25,
(pts[topRightCount].y+pts[topLeftCount].y)/2  - 10);
Point rigPos = Point((pts[topRightCount].x+pts[botRightCount].x)/2 + 1,
(pts[topRightCount].y+pts[botRightCount].y)/2);
Point botPos = Point((pts[botRightCount].x+pts[botLeftCount].x)/2 - 25,
(pts[botRightCount].y+pts[botLeftCount].y)/2 + 25);
Point lefPos = Point((pts[topLeftCount].x+pts[botLeftCount].x)/2 - 40,
(pts[topRightCount].y+pts[botLeftCount].y)/2-40);
Point areaPos(20, 340);
Point listPos(150, 340);
Point trimPos(280, 340);
Point skewPos(20, 310);
Point centerRefPos = Point(centerRef.x-70, centerRef.y);
Point centerDifPos = Point((center.x+centerRef.x)/2 - 10,
(center.y+centerRef.y)/2 + 20);
char str[20];
sprintf(str,"Area: %d", area);
char listStr[20];
sprintf(listStr,"List: %0.3f", listRate);
char trimStr[20];
sprintf(trimStr,"Trim: %0.3f", trimRate);
char centRef[20];
sprintf(centRef,"%d,%d", centerRef.x, centerRef.y);
char skewStr[20];
sprintf(skewStr,"Skew: %0.1f Degree", skewRate);
char centDif[20];
sprintf(centDif,"%d", lengthDif);
char str1[20];
sprintf(str1,"%d", topL);
char str2[20];
sprintf(str2,"%d", rigL);
char str3[20];
sprintf(str3,"%d", botL);
char str4[20];
sprintf(str4,"%d", lefL);
char coor[20];
sprintf(coor,"%d,%d", center.x, center.y);
char coor1[20];
sprintf(coor1,"%d,%d", pts[topLeftCount].x,  pts[topLeftCount].y);
char coor2[20];
sprintf(coor2,"%d,%d", pts[botRightCount].x, pts[botRightCount].y);
char coor3[20];
sprintf(coor3,"%d,%d", pts[topRightCount].x, pts[topRightCount].y);
char coor4[20];
sprintf(coor4,"%d,%d", pts[botLeftCount].x, pts[botLeftCount].y);
putText(imgAsli, str, areaPos, FONT_HERSHEY_SIMPLEX, 0.6, Scalar( 255, 0,
0), 2);
putText(imgAsli, listStr, listPos, FONT HERSHEY SIMPLEX, 0.6, Scalar( 255,
0, 0), 2);
putText(imgAsli, trimStr, trimPos, FONT_HERSHEY_SIMPLEX, 0.6, Scalar( 255,
0, 0), 2);
putText(imgAsli, skewStr, skewPos, FONT_HERSHEY_SIMPLEX, 0.6, Scalar( 255,
0, 0), 2);
putText(imgAsli, centRef, centerRefPos, FONT_HERSHEY_SIMPLEX, 0.5,
Scalar( 255, 0, 0), 2);
putText(imgAsli, centDif, centerDifPos, FONT_HERSHEY_SIMPLEX, 0.5,
Scalar( 0, 0, 255), 2);
putText(imgAsli, str1, topPos, FONT_HERSHEY_SIMPLEX, 0.6, Scalar( 0, 255,
0), 2);
putText(imgAsli, str2, rigPos, FONT_HERSHEY_SIMPLEX, 0.6, Scalar( 0, 255,
0), 2);
```

```
putText(imgAsli, str3, botPos, FONT_HERSHEY_SIMPLEX, 0.6, Scalar( 0, 255,
0), 2);
putText(imgAsli, str4, lefPos, FONT_HERSHEY_SIMPLEX, 0.6, Scalar( 0, 255,
0), 2);
putText(imgAsli, coor, Point(center.x+8, center.y+3), FONT_HERSHEY_SIMPLEX,
0.5, Scalar( 255, 0, 0), 2);
putText(imgAsli, coor1, Point(pts[topLeftCount].x-20, pts[topLeftCount].y-
10), FONT_HERSHEY_SIMPLEX, 0.45, Scalar( 0, 0, 255), 2);
putText(imgAsli, coor2, Point(pts[botRightCount].x-20,
pts[botRightCount].y+15), FONT_HERSHEY_SIMPLEX, 0.45, Scalar( 0, 0, 255),
2);
putText(imgAsli, coor3, Point(pts[topRightCount].x-20,
pts[topRightCount].y-10), FONT_HERSHEY_SIMPLEX, 0.45, Scalar( 0, 0, 255),
2);
putText(imgAsli, coor4, Point(pts[botLeftCount].x-20,
pts[botLeftCount].y+15), FONT_HERSHEY_SIMPLEX, 0.45, Scalar( 0, 0, 255),
2);
convexHull( pts, result );
return result;
}
static void onTrackbar(int, void*)
{
imgAsli = imread( tempImage[1], 1);
esize(imgAsli, imgAsli, Size(480, 320), 0, 0, INTER_CUBIC);
cvtColor(imgAsli, imgHSV, COLOR_RGB2HSV);
inRange(imgHSV, Scalar(H_Min, S_Min, V_Min), Scalar(H_Max, S_Max, V_Max),
imgBiner); //Threshold the tempImagege
medianBlur(imgBiner, imgBiner, 2*GB1+1);
    // GaussianBlur(imgBiner, imgBiner, Size(GB2*2+1, GB2*2+1), 0, 0);
    // bilateralFilter(imgBiner, imgBiner, GB2*2+1, (GB2*2+1)*2,
(GB2*2+1)/2);
erode(imgBiner, imgBiner, getStructuringElement(MORPH_RECT, Size(2*Erode+1,
2*Erode+1)) );
dilate( imgBiner, imgBiner, getStructuringElement(MORPH_RECT,
Size(2*Dilate+1, 2*Dilate+1)) );
dilate( imgBiner, imgBiner, getStructuringElement(MORPH_RECT,
Size(2*Dilate+1, 2*Dilate+1)) );
erode(imgBiner, imgBiner, getStructuringElement(MORPH_RECT, Size(2*Erode+1,
2*Erode+1)) );
medianBlur(imgBiner, imgBiner, 2*GB2+1);
vector<vector<Point> > contours;
vector<Vec4i> hierarchy;
Mat imgBiner2 = imgBiner.clone();
findContours( imgBiner2, contours, hierarchy, CV RETR EXTERNAL,
CV_CHAIN_APPROX_NONE, Point(0, 0) );
Mat drawing = Mat::zeros( imgBiner2.size(), CV_8UC3 );
for (int i = 0; i< contours.size(); i++)
{ Scalar color = Scalar( 255,255,255);
drawContours( drawing, contours, i, color, 2 );
}
vector<Point> ConvexHullPoints =  contoursConvexHull(contours);
polylines( drawing, ConvexHullPoints, true, Scalar(0,0,255), 2 );
imshow("Contours", drawing);
polylines( imgAsli, ConvexHullPoints, true, Scalar(0,0,255), 2 );
imshow("contoursConvexHull", src);
namedWindow("Biner 1", WINDOW_NORMAL);
namedWindow("Biner 2", WINDOW_NORMAL);
namedWindow("Asli", WINDOW_NORMAL);
resizeWindow("Asli", 480,320);
resizeWindow("Biner 1", 480,320);
resizeWindow("Biner 2", 480,320);
imshow("Asli", imgAsli);
imshow("Biner 1", imgBiner);
imshow("Biner 2", drawing);
```

```
}
void trackbar(){
createTrackbar("H_Min", "Control", &H_Min, 179, onTrackbar); //Hue (0 -
179)
createTrackbar("H_Max", "Control", &H_Max, 179, onTrackbar);
createTrackbar("S_Min", "Control", &S_Min, 255, onTrackbar); //Saturation
(0 - 255)
createTrackbar("S_Max", "Control", &S_Max, 255, onTrackbar);
createTrackbar("V_Min", "Control", &V_Min, 255, onTrackbar); //Value (0 -
255)
createTrackbar("V_Max", "Control", &V_Max, 255, onTrackbar);
createTrackbar("Erode", "Control", &Erode, 50, onTrackbar); //Value (0 -
255)
createTrackbar("Dilate", "Control", &Dilate, 50, onTrackbar);
createTrackbar("GB1", "Control", &GB1, 50, onTrackbar); //Value (0 - 255)
createTrackbar("GB2", "Control", &GB2, 50, onTrackbar);
}
int main( int argc, char** argv )
{
if ( argc != 2 ){
printf("usage: DisplaytempImagege.out <tempImagege_Path>\n");
return -1;
}
namedWindow("Control", WINDOW_NORMAL);
trackbar();
tempImage = argv;
onTrackbar(0, 0);
waitKey(0)==!27;
return 0;
}
```

## BIODATA PENULIS

Penulis lahir pada tanggal 7 Agustus 1995 di Tangerang, Banten, anak pertama dari pasangan Sohib Romdhoni dan Luluk Alifah. Pendidikan yang ditempuh antara lain TK Al-Fathir, Tangerang (2000-2001), lalu melanjutkan ke SDN Sukasari 4 Tangerang (2001-2004), lalu pindah ke SDN Baureno 1, Bojonegoro (2004-2007), kemudian ke SMP Plus Ar-Rahmat Bojonegoro (2007-2010), selanjutnya penulis melanjutkan ke SMA Negeri Model Terpadu Bojonegoro (2010-2013). Penulis kemudian melanjutkan pendidikan ke perguruan tinggi di Insitut Teknologi Sepuluh Nopember, Departemen Teknik Sistem Perkapalan (2013-2017). Selama menjadi mahasiswa di ITS, penulis aktif dalam kegiatan UKM Maritime Challenge (2013), Marine Technology and Innovation Club (2014), LDJ Al-Mi'raj Teknik Sistem Perkapalan (2014), Tim Barunastra Roboboat ITS (2014-2016), UKM Robotika ITS (2015), dan Asisten Laboratorium Marine Electrical and Automation System (2016-2017) pada praktikum dasar digital dan *Automatic Change Over Switch*. Sebagai anggota tim Barunastra Roboboat ITS yang bekerja di bidang *programming* khususnya mengenai pengolahan citra, penulis bersama dengan tim, berpartisipasi dalam lomba KKCTBN (2014) di Universitas Indonesia, Deconbotion (2015) di Universitas Diponegoro, dan *9$^{th}$ AUVSI Roboboat Competition* (2016) di Virginia Beach, VA, USA. Tim termasuk penulis mendapatkan penghargaan *Best Design* pada KKCTBN, Juara 1 pada Deconbotion, Juara 3 dan penghargaan *Savitsky Award* pada *9$^{th}$ AUVSI Roboboat Competition*.